Open Financial Exchange Specification 1.5

April 9, 1998

© 1997, 1998 CheckFree Corp., Intuit Inc., Microsoft Corp. All rights reserved

Open Financial Exchange Specification Legend

Open Financial Exchange Specification ©1996-98 by its publishers: CheckFree Corp., Intuit Inc., and Microsoft Corporation. All rights reserved.
A royalty-free, worldwide, and perpetual license is hereby granted to any party to use the Open Financial Exchange Specification to make, use, and sell products and services that conform to this Specification.
THIS OPEN FINANCIAL EXCHANGE SPECIFICATION IS MADE AVAILABLE “AS IS” WITHOUT WARRANTY OF ANY KIND. TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, MICROSOFT, INTUIT AND CHECKFREE (“PUBLISHERS”) FURTHER DISCLAIM ALL WARRANTIES, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT, ALL OF WHICH ARE HEREBY DISCLAIMED. THE ENTIRE RISK ARISING OUT OF THE USE OF THIS SPECIFICATION REMAINS WITH RECIPIENT. TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, IN NO EVENT SHALL THE PUBLISHERS OF THIS SPECIFICATION BE LIABLE FOR ANY CONSEQUENTIAL, INCIDENTAL, DIRECT, INDIRECT, SPECIAL, PUNITIVE, OR OTHER DAMAGES WHATSOEVER (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION, OR OTHER PECUNIARY LOSS) ARISING OUT OF ANY USE TO WHICH THIS SPECIFICATION IS PUT, EVEN IF THE PUBLISHERS HEREOF HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Contents

1. OVERVIEW....1

1.1 INTRODUCTION....1
1.1.1 Design Principles....1
1.2 OPEN FINANCIAL EXCHANGE AT A GLANCE....2
1.2.1 Data Transport....2
1.2.2 Request and Response Model....3
1.3 CONVENTIONS....4

2. STRUCTURE....7

2.1 HTTP HEADERS....8
2.2 OPEN FINANCIAL EXCHANGE HEADERS....9
2.2.1 OFXHEADER....10
2.2.2 DATA....10
2.2.3 VERSION....10
2.2.4 SECURITY....11
2.2.5 ENCODING and CHARSET....11
2.2.6 COMPRESSION....11
2.2.7 OLDFILEUID and NEWFILEUID....11
2.3 SGML DETAILS....12
2.3.1 Compliance....12
2.3.2 Valid SGML Characters....12
2.3.3 Comments Not Supported....13
2.4 OPEN FINANCIAL EXCHANGE SGML STRUCTURE....13
2.4.1 Overview....13
2.4.2 Case Sensitivity....13
2.4.3 Top Level....13
2.4.4 Messages....13
2.4.5 Message Sets and Version Control....14
2.4.6 Transactions....16
2.5 THE SIGNON MESSAGE SET....18
2.5.1 Signon <SONRQ> <SONRS>....18
2.5.2 USERPASS Change <PINCHRQ> <PINCHRS>....21
2.5.3 Signon Message Set Profile Information....23
2.5.4 Examples....23
2.6 EXTERNAL DATA SUPPORT....23
2.7 EXTENSIONS TO OPEN FINANCIAL EXCHANGE....24

3. COMMON AGGREGATES, ELEMENTS, AND DATA TYPES....25

3.1 COMMON AGGREGATES....25
3.1.1 Identification of Financial Institutions and Accounts....25
3.1.2 Format of User-Supplied Numbers....25
3.1.3 Balance Records <BAL>....25
3.1.4 Error Reporting <STATUS>....26
3.2 COMMON ELEMENTS....27
3.2.1 Financial Institution Transaction ID <FITID>....27
3.2.2 Server-Assigned ID <SRVRTID>, <SRVRTID2>....28
3.2.3 Client-Assigned Transaction UID <TRNUID>....28
3.2.4 Token <TOKEN>, <TOKEN2>....28
3.2.5 Transaction Amount <TRNAMT>....28
3.2.6 Memo <MEMO>, <MEMO2>....29
3.2.7 Date Start and Date End <DTSTART> <DTEND>....29
3.2.8 Common Data Types....30
3.2.9 Amounts, Prices, and Quantities....31
3.2.10 Language....32

3.2.11 Other Basic Data Types....32

4. OPEN FINANCIAL EXCHANGE SECURITY....33

4.1 SECURITY CONCEPTS IN OFX....33
4.1.1 Architecture....33
4.1.2 Security Goals....33
4.1.3 Security Standards....34
4.1.4 FI Responsibilities....34
4.1.5 Security Levels: Channel vs. Application....35
4.2 SECURITY IMPLEMENTATION IN OFX....36
4.2.1 Channel-Level Security....36
4.2.2 Application-Level Security....37

5. INTERNATIONAL SUPPORT....43

5.1 LANGUAGE AND ENCODING....43
5.2 CURRENCY <CURDEF> <CURRENCY> <ORIGCURRENCY>....43
5.3 COUNTRY-SPECIFIC TAG VALUES....44

6. DATA SYNCHRONIZATION....45

6.1 OVERVIEW....45
6.2 BACKGROUND....45
6.3 DATA SYNCHRONIZATION APPROACH....46
6.4 DATA SYNCHRONIZATION SPECIFICS....47
6.5 CONFLICT DETECTION AND RESOLUTION....49
6.6 SYNCHRONIZATION VS. REFRESH....49
6.7 TYPICAL SERVER ARCHITECTURE FOR SYNCHRONIZATION....51
6.8 TYPICAL CLIENT PROCESSING OF SYNCHRONIZATION RESULTS....52
6.9 SIMULTANEOUS CONNECTIONS....52
6.10 SYNCHRONIZATION ALTERNATIVES....53
6.10.1 Lite Synchronization....53
6.10.2 Relating Synchronization and Error Recovery....54
6.11 EXAMPLES....54

7. FI PROFILE....57

7.1 OVERVIEW....57
7.1.1 Message Sets....58
7.1.2 Version Control....58
7.1.3 Batching and Routing....58
7.1.4 Client Signon for Profile Requests....59
7.1.5 Profile Request <PROFRQ>....59
7.2 PROFILE RESPONSE <PROFRS>....60
7.2.1 Message Set....61
7.2.2 Signon Realms....62
7.2.3 Status Codes....63
7.3 PROFILE MESSAGE SET PROFILE INFORMATION....63

8. ACTIVATION & ACCOUNT INFORMATION....65

8.1 OVERVIEW....65
8.2 APPROACHES TO USER SIGN-UP WITH OPEN FINANCIAL EXCHANGE....65
8.3 USERS AND ACCOUNTS....66
8.4 ENROLLMENT AND PASSWORD ACQUISITION....66
8.4.1 User IDs....67
8.4.2 Enrollment Request <ENROLLRQ>....67
8.4.3 Enrollment Response <ENROLLRS>....68
8.4.4 Enrollment Status Codes....68
8.4.5 Examples....68
8.5 ACCOUNT INFORMATION....70
8.5.1 Request <ACCTINFORQ>....70

8.5.2 Response <ACCTINFORS>....70
8.5.3 Account Information Aggregate <ACCTINFO>....71
8.5.4 Status Codes....71
8.5.5 Examples....71
8.6 SERVICE ACTIVATION....72
8.6.1 Activation Request and Response....72
8.6.2 Service Activation Synchronization....75
8.6.3 Examples....75
8.7 NAME AND ADDRESS CHANGES....76
8.7.1 Change User Information Request <CHGUSERINFORQ>....76
8.7.2 Change User Information Response <CHGUSERINFORS>....76
8.7.3 Status Codes....77
8.8 SIGNUP MESSAGE SET PROFILE INFORMATION....77

9. CUSTOMER TO FI COMMUNICATION....79

9.1 THE E-MAIL MESSAGE SET....79
9.2 E-MAIL MESSAGES....79
9.2.1 Regular vs. Specialized E-Mail....79
9.2.2 Basic <MAIL> Aggregate....79
9.2.3 E-Mail <MAILRQ> <MAILRS>....81
9.2.4 E-Mail Synchronization <MAILSYNCRQ> <MAILSYNCRS>....81
9.2.5 E-Mail Example....82
9.3 GET HTML PAGE....84
9.3.1 MIME Get Request and Response <GETMIMERQ> <GETMIMERS>....84
9.3.2 MIME Example....85
9.4 E-MAIL MESSAGE SET PROFILE INFORMATION....85

10. RECURRING TRANSACTIONS....87

10.1 CREATING A RECURRING MODEL....87
10.2 RECURRING INSTRUCTIONS <RECURRINST>....87
10.2.1 Values for <FREQ>....87
10.2.2 Examples....88
10.3 RETRIEVING TRANSACTIONS GENERATED BY A RECURRING MODEL....89
10.4 MODIFYING AND CANCELING INDIVIDUAL TRANSACTIONS....89
10.5 MODIFYING AND CANCELING RECURRING MODELS....90
10.5.1 Examples....90

11. BANKING....93

11.1 CONSUMER AND BUSINESS BANKING....93
11.2 CREDIT CARD DATA....93
11.3 COMMON BANKING AGGREGATES....93
11.3.1 Banking Account <BANKACCTFROM> and <BANKACCTTO>....93
11.3.2 Credit Card Account <CCACCTFROM>....97
11.3.3 Bank Account Information <BANKACCTINFO>....97
11.3.4 Credit Card Account Information <CCACCTINFO>....97
11.3.5 Transfer Information <XFERINFO>....98
11.3.6 Transfer Processing Status <XFERPRCSTS>....99
11.4 DOWNLOADING TRANSACTIONS AND BALANCES....100
11.4.1 Bank Statement Download....100
11.4.2 Credit Card Statement Download....102
11.4.3 Statement Transaction <STMTTRN>....103
11.5 STATEMENT CLOSING INFORMATION....105
11.5.1 Statement Closing Download....106
11.5.2 Non-Credit Card Statement <CLOSING>....107
11.5.3 Credit Card Statement Closing Request <CCSTMTENDRQ>....108
11.5.4 Credit Card Statement Closing Response <CCSTMTENDRS>....108
11.6 STOP CHECK....110
11.6.1 Stop Check Add....110

11.6.2 Status Codes....112
11.7 INTRABANK FUNDS TRANSFER....113
11.7.1 Intrabank Funds Transfer Addition....113
11.7.2 Intrabank Funds Transfer Modification....115
11.7.3 Intrabank Funds Transfer Cancellation....116
11.8 INTERBANK FUNDS TRANSFER....117
11.8.1 Interbank Funds Transfer – US....117
11.8.2 Interbank Funds Transfer – International Usage....118
11.8.3 Interbank Funds Transfer Modification....119
11.8.4 Interbank Funds Transfer Cancellation....121
11.8.5 Multiple Interbank Funds Transfer....122
11.9 WIRE FUNDS TRANSFER....123
11.9.1 Wire Funds Transfer Addition....124
11.9.2 Wire Funds Transfer Cancellation....126
11.10 RECURRING FUNDS TRANSFER....127
11.10.1 Recurring Intrabank Funds Transfer Addition....127
11.10.2 Recurring Intrabank Funds Transfer Modification....129
11.10.3 Recurring Intrabank Funds Transfer Cancellation....131
11.10.4 Recurring Interbank Funds Transfer Addition....132
11.10.5 Recurring Interbank Funds Transfer Modification....134
11.10.6 Recurring Interbank Funds Transfer Cancellation....135
11.11 E-MAIL AND CUSTOMER NOTIFICATION....136
11.11.1 Banking E-Mail....136
11.11.2 Notifications....138
11.11.3 Returned Check and Deposit Notification....138
11.12 DATA SYNCHRONIZATION FOR BANKING....139
11.12.1 Data Synchronization for Stop Check....139
11.12.2 Data Synchronization for Intrabank Funds Transfers....140
11.12.3 Data Synchronization for Interbank Funds Transfers....142
11.12.4 Data Synchronization for Wire Funds Transfers....143
11.12.5 Data Synchronization for Recurring Intrabank Funds Transfers....144
11.12.6 Data Synchronization for Recurring Interbank Funds Transfers....145
11.12.7 Data Synchronization for Bank Mail....147
11.12.8 Status Codes....148
11.13 MESSAGE SETS AND PROFILE....148
11.13.1 Message Sets and Messages....148
11.13.2 Bank Message Set Profile....155
11.13.3 Credit Card Message Set Profile....157
11.13.4 Interbank Funds Transfer Message Set Profile....157
11.13.5 Wire Transfer Message Set Profile....158
11.14 EXAMPLES....159
11.14.1 Statement Download....159
11.14.2 Intrabank Funds Transfer....160
11.14.3 Stop Check....162
11.14.4 Recurring Transfers....163

12. PAYMENTS....171

12.1 CONSUMER AND BUSINESS PAYMENTS....171
12.2 THE PAYEE MODEL....171
12.2.1 Payee Identifiers....171
12.2.2 Payee Lists....172
12.2.3 Standard Payee Lists....172
12.2.4 Summary – Identifying Payees....173
12.3 IDENTIFIERS USED IN PAYMENT TRANSACTIONS....174
12.4 THE PAYMENT LIFE CYCLE....174
12.4.1 Payment Creation....174
12.4.2 Payment Modification....175
12.4.3 Payment Status Inquiry....176

12.4.4 Payment Cancellation....176
12.4.5 Delayed Payee Matching....176
12.5 COMMON PAYMENTS AGGREGATES....176
12.5.1 Payments Account Information <BPACCTINFO>....176
12.5.2 Payment Information <PMTINFO>, <PMTINFO2>....177
12.6 PAYMENTS FUNCTIONS....183
12.6.1 Payment Creation....184
12.6.2 Payment Modification....186
12.6.3 Payment Cancellation....188
12.6.4 Payment Status Inquiry....189
12.7 RECURRING PAYMENTS....190
12.7.1 Creating a Recurring Payment....191
12.7.2 Recurring Payment Modification....193
12.7.3 Recurring Payment Cancellation....195
12.8 PAYMENT MAIL....196
12.8.1 Payment Mail Request and Response....196
12.8.2 Payment Mail Synchronization....197
12.9 PAYEE LISTS....198
12.9.1 Adding a Payee to the Payee List....199
12.9.2 Payee Modification....201
12.9.3 Payee Deletion....203
12.9.4 Payee List Synchronization....204
12.10 DATA SYNCHRONIZATION FOR PAYMENTS....205
12.10.1 Payment Synchronization....205
12.10.2 Recurring Payment Synchronization....206
12.10.3 Discussion....208
12.11 MESSAGE SETS AND PROFILE....209
12.11.1 Bill Pay Message Sets and Messages....209
12.11.2 Bill Pay Message Set Profile <BILLPAYMSGSET>....211
12.12 EXAMPLES....214
12.12.1 Scheduling a Payment....214
12.12.2 Modifying a Payment....217
12.12.3 Canceling a Payment....220
12.12.4 Updating Payment Status....221
12.12.5 Scheduling a Recurring Payment....222
12.12.6 Modifying a Recurring Payment....223
12.12.7 Canceling a Recurring Payment....225
12.12.8 Adding a Payee to the Payee List....226
12.12.9 Synchronizing Scheduled Payments....227

13. INVESTMENTS....231

13.1 TYPES OF RESPONSE INFORMATION....231
13.2 SUB-ACCOUNTS....232
13.3 UNITS, PRECISION, AND SIGNS....232
13.3.1 Units....232
13.3.2 Precision....232
13.3.3 Signs....233
13.4 BANK AND INVESTMENT TRANSACTIONS....233
13.5 MONEY MARKET FUNDS....233
13.5.1 Separate Account at the Financial Institution....234
13.5.2 Sweep Account Within an Investment Account....234
13.5.3 Position Within an Investment Account....234
13.6 INVESTMENT ACCOUNTS....234
13.6.1 Specifying the Investment Account <INVACCTFROM>....234
13.6.2 Investment Account Information <INVACCTINFO>....235
13.6.3 Brokerage, Mutual Fund, and 401K Accounts....236
13.7 INVESTMENT MESSAGE SETS AND PROFILE....236
13.7.1 Investment Statement Download....236

13.7.2 Security Information....238
13.8 INVESTMENT SECURITIES....239
13.8.1 Security Identification <SECID>....239
13.8.2 Security List Request....240
13.8.3 Security List Response....241
13.8.4 Security List <SECLIST>....241
13.8.5 Securities Information....242
13.9 INVESTMENT STATEMENT DOWNLOAD....245
13.9.1 Investment Statement Request....245
13.9.2 Investment Statement Response....246
13.10 INVESTMENT E-MAIL....258
13.10.1 Investment E-Mail Request and Response....258
13.10.2 Investment E-Mail Synchronization....259
13.11 COMPLETE EXAMPLE....260

14. BILL PRESENTMENT....264

14.1 OVERVIEW....264
14.1.1 Bill Presentment Model....264
14.1.2 Servers and Message Sets....264
14.2 BILLER DIRECTORY....265
14.2.1 Client Signon to the Biller Directory Server....265
14.2.2 Search Arguments....265
14.2.3 Identification of Bill Publishers....265
14.2.4 Find Biller Request <FINDBILLERRQ>....265
14.2.5 Find Biller Response <FINDBILLERRS>....266
14.2.6 Status Codes <FINDBILLERTRNRS>....267
14.2.7 Account Number Validation....267
14.3 CUSTOMER SIGNUP....269
14.3.1 Enrollment....269
14.3.2 Account Inquiry....269
14.3.3 Service Activation....271
14.3.4 Service Status Update for Groups of Customers....271
14.3.5 Biller Payment Restrictions....273
14.4 BILL DELIVERY....275
14.4.1 Bill Delivery Process....275
14.4.2 Bill List Retrieval....275
14.4.3 Bill Detail Retrieval....279
14.4.4 Table Structure Definition....282
14.4.5 Delivery Notification....283
14.5 BILL PAYMENT....284
14.5.1 Remittance Information....284
14.5.2 Payee Identification....284
14.6 BILL PRESENTMENT E-MAIL....284
14.6.1 Bill Presentment Mail Request <PRESMAILRQ>....285
14.6.2 Bill Presentment Mail Response <PRESMAILRS>....285
14.6.3 Status Codes <PRESMAILTRNRS>....285
14.7 MESSAGE SETS AND PROFILE....286
14.7.1 Message Sets and Messages....286
14.7.2 Biller Directory Message Set Profile....288
14.7.3 Bill Delivery Message Set Profile....288
14.8 BILL PRESENTMENT EXAMPLES....289
14.8.1 Find Biller Examples....289
14.8.2 Enrollment Examples....291
14.8.3 Activation Example....292
14.8.4 Bill Delivery Examples....294

A. STATUS CODES....A-1

B. CHANGE HISTORY....B-4

B.1 OFX 1.0 TO 1.0.1 ..... B-4
B.1.1 Specification Changes by Chapter B-4
B.1.2 General Specification Changes B-17
B.1.3 DTD Changes B-17
B.2 OFX 1.0.1 TO 1.0.2 ..... B-20
B.2.1 Specification Changes by Chapter B-20
B.2.2 General Specification Changes B-23
B.2.3 DTD Changes B-23

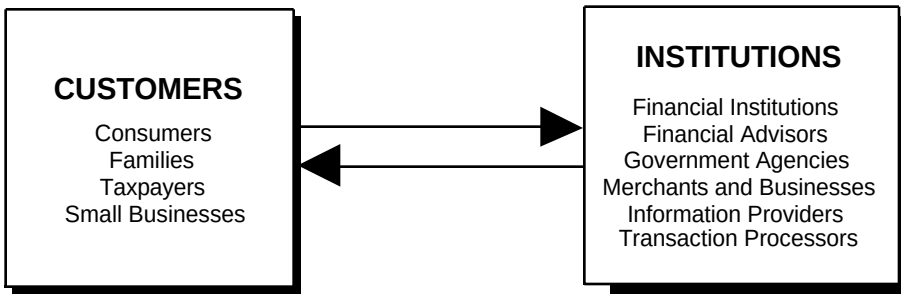
C. ERRORS AND OMISSIONS C-1

D. TAG INDEX D-1

1. Overview

1.1 Introduction

Open Financial Exchange is a broad-based framework for exchanging financial data and instructions between customers and their financial institutions. It allows institutions to connect directly to their customers without requiring an intermediary.



Open Financial Exchange is an open specification that anyone can implement: any financial institution, transaction processor, software developer, or other party. It uses widely accepted open standards for data formatting (such as SGML), connectivity (such as TCP/IP and HTTP), and security (such as SSL).

Open Financial Exchange defines the request and response messages used by each financial service as well as the common framework and infrastructure to support the communication of those messages. This specification does not describe any specific product implementation.

1.1.1 Design Principles

The following principles were used in designing Open Financial Exchange:

- **Broad Range of Financial Activities** – Open Financial Exchange provides support for a ***broad*** range of financial activities. Open Financial Exchange 1.5 specifies the following services:
  - Bank statement download
  - Credit card statement download
  - Funds transfers including recurring transfers
  - Consumer payments, including recurring payments
  - Business payments, including recurring payments
  - Brokerage and mutual fund statement download, including transaction history, current holdings, and balances.
  - Bill presentment and payment
- **Broad Range of Financial Institutions** – Open Financial Exchange supports communication with a ***broad*** range of financial institutions (FIs), including:
  - Banks
  - Brokerage houses
  - Merchants
  - Processors
  - Financial advisors
  - Government agencies
- **Broad Range of Front-End Applications** – Open Financial Exchange supports a ***broad*** range of front-end applications, including Web-based applications, covering all types of financial activities running on all types of platforms.
- **Extensible** – Open Financial Exchange has been designed to allow the easy addition of new services. Future versions will include support for many new services.
- **Open** – This specification is publicly available. You can build client and server applications using the Open Financial Exchange protocols independent of any specific technology, product, or company.

- **Multiple Client Support** – Open Financial Exchange allows a user to use multiple client applications to access the same data at a financial institution. With the popularity of the World Wide Web, customers are increasingly more likely to use multiple applications—either desktop-based or Web-based—to perform financial activities. For example, a customer can track personal finances at home with a desktop application and occasionally pay bills while at work with a Web-based application. The use of data synchronization to support multiple clients is a key innovation in Open Financial Exchange.
- **Robust** – Open Financial Exchange will be used for executing important financial transactions and for communicating important financial information. Assuring users that transactions are executed and information is correct is crucial. Open Financial Exchange provides robust protocols for error recovery.
- **Secure** – Open Financial Exchange provides a framework for building secure online financial services. In Open Financial Exchange, security encompasses authentication of the parties involved, as well as secrecy and integrity of the information being exchanged.
- **Batch & Interactive** – The design of request and response messages in Open Financial Exchange is for use in either batch or interactive style of communication. Open Financial Exchange provides for applying a single authentication context to multiple requests in order to reduce the overhead of user authentication.
- **International Support** – Open Financial Exchange is designed to supply financial services throughout the world. It supports multiple currencies, country-specific extensions, and different forms of encoding such as UNICODE.
- **Platform Independent** –Open Financial Exchange can be implemented on a wide variety of front-end client devices, including those running Windows 3.1, Windows 95, Windows NT, Macintosh, or UNIX. It also supports a wide variety of Web-based environments, including those using HTML, Java, JavaScript, or ActiveX. Similarly on the back-end, Open Financial Exchange can be implemented on a wide variety of server systems, including those running UNIX, Windows NT, or OS/2.
- **Transport Independent** – Open Financial Exchange is independent of the data communication protocol used to transport the messages between the client and server computers. Open Financial Exchange 1.0.2 and 1.5 will use HTTP.

1.2 Open Financial Exchange at a Glance

The design of Open Financial Exchange is as a client and server system. An end-user uses a client application to communicate with a server at a financial institution. The form of communication is requests from the client to the server and responses from the server back to the client.

Open Financial Exchange uses the Internet Protocol (IP) suite to provide the communication channel between a client and a server. IP protocols are the foundation of the public Internet and a private network can also use them.

1.2.1 Data Transport

Clients use the HyperText Transport Protocol (HTTP) to communicate to an Open Financial Exchange server. The World Wide Web throughout uses the same HTTP protocol. In principle, a financial institution can use any off-the-shelf web server to implement its support for Open Financial Exchange.

To communicate by means of Open Financial Exchange over the Internet, the client must establish an Internet connection. This connection can be a dial-up Point-to-Point Protocol (PPP) connection to an Internet Service Provider (ISP) or a connection over a local area network that has a gateway to the Internet.

Clients use the HTTP POST command to send a request to the previously acquired Uniform Resource Locator (URL) for the desired financial institution. The URL presumably identifies a Common Gateway Interface (CGI) or other process on an FI server that can accept Open Financial Exchange requests and produce a response.

The POST identifies the data as being of type application/x-ofx. Use application/x-ofx as the return type as well. Fill in other fields per the HTTP 1.0 spec. Here is a typical request:

```
POST http://www.fi.com/ofx.cgi HTTP/1.0          HTTP headers
```

```
User-Agent:MyApp 5.0
Content-Type: application/x-ofx
Content-Length: 1032

OFXHEADER:100                                    OFX headers
DATA:OFXSGML
VERSION:150
SECURITY:TYPE1
ENCODING:USASCII

<OFX>                                            OFX request
... Open Financial Exchange requests ...
</OFX>
```

A blank line defines the separation between the HTTP headers and the start of the Open Financial Exchange headers. A blank line also separates the Open Financial Exchange headers and the request. (See Chapter 2, “Structure” for more information about the Open Financial Exchange headers.) A brief note here that a “blank line” means a carriage return and a linefeed pair – CRLF.

The structure of a response is similar to the request, with the first line containing the standard HTTP result, as shown next. The content length is given in bytes.

```
HTTP 1.0 200 OK                                  HTTP headers
Content-Type: application/x-ofx
Content-Length: 8732

OFXHEADER:100                                    OFX headers
DATA:OFXSGML
VERSION:150
SECURITY:TYPE1
ENCODING:USASCII

<OFX>                                            OFX response
... Open Financial Exchange responses ...
</OFX>
```

1.2.2 Request and Response Model

The basis for Open Financial Exchange is the request and response model. One or more requests can be batched in a single file. This file typically includes a signon request and one or more service-specific requests. An FI server will process all of the requests and return a single response file. This batch model lends itself to Internet transport as well as other off-line transports. Both requests and responses are plain text files, formatted using a grammar based on Standard Generalized Markup Language (SGML). Open Financial Exchange is syntactically similar to HyperText Markup Language (HTML), featuring tags to identify and delimit the data. The use of a tagged data format allows Open Financial Exchange to evolve over time while continuing to support older clients and servers.

Here is a simplified example of an Open Financial Exchange request file. (This example does not show the Open Financial Exchange headers and the indentation is only for readability.) For complete details, see the more complete examples throughout this specification.

```
<OFX>                                     <!-- Begin request data -->
   <SIGNONMSGSRQV1>
      <SONRQ>                             <!-- Begin signon -->
         <DTCLIENT>19961029101000         <!-- Oct. 29, 1996, 10:10:00 am -->
         <USERID>123-45-6789              <!-- User ID (that is, SSN) -->
         <USERPASS>MyPassword             <!-- Password (SSL encrypts whole) -->
         <LANGUAGE>ENG                    <!-- Language used for text -->
         <FI>                             <!-- ID of receiving institution -->
            <ORG>NCH <!-- Name of ID owner -->
            <FID>1001                     <!-- Actual ID -->
         </FI>
         <APPID>MyApp
```

```
        <APPVER>0500
      </SONRQ>      <!-- End of signon -->
   </SIGNONMSGSRQV1>

   <BANKMSGSRQV1>
      <STMTTRNRQ>   <!-- First request in file -->
         <TRNUID>1001
         <STMTRQ>   <!-- Begin statement request -->
            <BANKACCTFROM>                 <!-- Identify the account -->
               <BANKID>121099999           <!-- Routing transit or other FI ID -->
               <ACCTID>999988              <!-- Account number -->
               <ACCTTYPE>CHECKING          <!-- Account type -->
               </BANKACCTFROM>             <!-- End of account ID -->
            <INCTRAN>                      <!-- Begin include transaction -->
               <INCLUDE>Y                  <!-- Include transactions -->
            </INCTRAN>                     <!-- End of include transaction -->
         </STMTRQ>                         <!-- End of statement request -->
      </STMTTRNRQ>                         <!-- End of first request -->
   </BANKMSGSRQV1>
</OFX>                                     <!-- End of request data -->
```

The response format follows a similar structure. Although a response such as a statement response contains all of the details of each transaction, each element is identified using tags.

The key rule of Open Financial Exchange syntax is that each tag is either an element or an aggregate. Data follows its element tag. An aggregate tag begins a compound tag sequence, which must end with a matching tag; for example, <AGGREGATE> ... </AGGREGATE>.

The file sent by Open Financial Exchange does not require any white space between tags.

White space following a tag delimiter (>), following an element value, or preceding a tag delimiter (<) should be ignored. White space within an element value (i.e. not preceding, not following) is significant. If white space is desired preceding or following an element value, this is achieved using the   macro. If more than one white space element is needed, then multiple   macros should be utilized.

1.3 Conventions

The conventions used in the tag descriptions include the following:

- Required tags are in **bold**. Regular face indicates tags that are optional. Required means that a client must always include the tag in a request, and a server must always include the tag in a response.
- Required tags occur once unless noted as one or more in the description, in which case the specification allows multiple occurrences.
- Optional tags occur once if present unless noted as zero or more in the description, in which case the specification allows multiple occurrences.
- A-*n* or N-*n* specify those values that take an alphanumeric or numeric type value, where *n* indicates the maximum size.
- Common value types, such as a dollar amount, are referenced by name. Chapter 3, “Common Aggregates, Elements, and Data Types,” lists value types that are referenced by name.
- In some aggregates, there are different tags used in different message set versions. In those aggregates, there is a third column in the table, called Version. In that column, for any tags that differ across message set versions, there is a description of the difference. For example, many tags say “V2 only” to indicate that they are in message set version 2 only.
- Explanatory information is in *italics*.

Tag	*Version*	*Description*
<REQUIREDTAG>		Required tag (1 or more)
<REQUIREDTAG2>		Required tag that occurs only once
<OPTIONALTAG>		Optional tag; this tag can occur multiple times (0 or more)
<SPECIFIC>		Values are A, B, and C

Tag	***Version***	***Description***
<ALPHAVALUE>		Takes an alphanumeric value up to 32 characters, ***A-32***
<NEWTAG>	V2 only	Tag used only in version 2 of this message set
Explanatory text		Hopefully useful information.

2. Structure

This chapter describes the basic structure of an Open Financial Exchange request and response. Structure includes headers, basic syntax, and the Signon request and response. This chapter also describes how Open Financial Exchange encodes external data, such as bit maps.

Open Financial Exchange data consists of some headers plus one Open Financial Exchange data block. This block consists of a signon message and zero or more additional messages. When sent over the Internet using HTTP, standard HTTP and multi-part MIME headers and formats surround the Open Financial Exchange data. A simple file that contained only Open Financial Exchange data would have the following form:

```
HTTP headers
MIME type application/x-ofx
Open Financial Exchange headers
Open Financial Exchange SGML block
```

A more complex file that contained additional Open Financial Exchange data would have this form:

```
HTTP headers
MIME type multipart/x-mixed-replace; boundary =--boundary--
---boundary---
MIME type application/x-ofx
   Open Financial Exchange headers
   Open Financial Exchange SGML block

---boundary---
   MIME type image/jpeg
      FI logo
```

Version 1.0.2 of the Open Financial Exchange specification did not specify how to properly separate the various components of an OFX request. In particular, separation of the HTTP headers, the MIME attachments, the OFX headers, the OFX header elements, and the OFX SGML block.

OFX 1.0.2 clients used a mix of LF and CRLF constructs and OFX 1.0.2 servers handled either linefeed (LF) or carriage return/line feed (CRLF), but not often both. In the future, it is expected that 1.0.2 servers will be upgraded to handle both CRLF and LF.

In version 1.5 of the Open Financial Exchange specification, the behavior for both an OFX client and an OFX server has been specified to encourage uniform usage of the specification.

OFX 1.5 Client

A proper client should separate the components of an OFX request using a single CRLF between each component. A proper request thus has the form:

HTTP headers
CRLF(s)
MIME type information
CRLF(s)
OFX header element 1
CRLF
OFX header element 2
CRLF
OFX header element *n*
CRLF(s)
OFX SGML Block

OFX 1.5 Server

An OFX 1.5 specification server should expect OFX request components and elements to be separated by the appropriate number of CRLF characters. However, as per W3C recommendations, an OFX 1.5 server should also accept just a LF as a separator. This behavior is as per the recommendation of the World Wide Web Consortium (W3C). The W3C is the worldwide standards body for web technology.

http://www.w3.org (W3C home page)

http://www.w3.org/Protocols/HTTP/OldClients.html (W3C recommendations)

The text has been included below for ease of reference:

Note: Server tolerance of bad clients

Whilst it is seen appropriate for testing parsers to check full conformance to this specification, it is recommended that operational parsers be tolerant of deviations.

In particular, lines should be regarded as terminated by the Line Feed, and the preceding Carriage Return character ignored.

Any HTTP Header Field Name which is not recognized should be ignored in operational parsers.

It is recommended that servers use URIs free of "variant" characters whose representation differs in some of the national variant character sets, punctuation characters, and spaces. This will make URIs easier to handle by humans when the need (such as debugging, or transmission through non-hypertext systems) arises.

Copyright © 1992, W3C.

2.1 HTTP Headers

Data delivered by way of HTTP places the standard HTTP result code on the first line. HTTP defines a number of status codes. Servers can return any standard HTTP result. However, FIs should expect clients to collapse these codes into the following three cases:

Code	Meaning	Action
200	OK	The request was processed and a valid Open Financial Exchange result is returned.
400s	Bad request	The request was invalid and was not processed. Clients will report an internal error to the user.
500s	Server error	The server is unavailable. Clients should advise the user to retry shortly.

Note: *The server must return code 400 for any problem that prevents it from processing the request file. Processing problems include failures relating to security, communication, parsing, or the Open Financial Exchange headers (for example, the client requested an unsupported language). For content errors such as wrong USERPASS or invalid account, the server must return a valid Open Financial Exchange response along with code 200. If a communication time-out error occurs while an OFX server and a back-end server are communicating to fill a request, then the server MUST return code 500.*

Open Financial Exchange requires the following HTTP standard headers:

Code	Value	Explanation
Content-type	application/x-ofx	The MIME type for Open Financial Exchange
Content-length	length	Length of the data after removing HTTP headers

When responding with multi-part MIME, the main type will be multi-part/x-mixed-replace; one of the parts will use application/x-ofx.

2.2 Open Financial Exchange Headers

The contents of an Open Financial Exchange file consist of a simple set of headers followed by contents defined by that header

The Open Financial Exchange headers are in a simple *tag:value* syntax and terminated by a blank line. Open Financial Exchange always sends headers unencrypted, even if application-level encryption is used for the remaining contents. The language and character set used for the headers is the same as the preceding MIME headers.

The first entry will always be OFXHEADER with a version number. This entry identifies the contents as an Open Financial Exchange file and provides the version number of the Open Financial Exchange headers that follow (not the version number of the contents). For example:

```
OFXHEADER:100
```

Open Financial Exchange headers can contain the following tags.

```
DATA:OFXSGML
VERSION:150
SECURITY:
ENCODING:
CHARSET:
COMPRESSION:
OLDFILEUID:
NEWFILEUID:
```

All OFX headers are required. NONE should be returned if client or server does not make use of an individual tag, e.g., COMPRESSION:NONE, OLDFILEUID:NONE

A blank line follows the last header. Then (for type OFXSGML), the SGML-readable data begins with the <OFX> tag.

For information about each of the OFX headers, refer to the following sections.

2.2.1 OFXHEADER

OFXHEADER specifies the version number of the Open Financial Exchange headers.

The OFXHEADER value should change its major number only if an existing client is unable to process the new header. This can occur because of a complete syntax change in a header, or a significant change in the semantics of an existing tag—not the entire response. A server can add new tags as long as clients can function without understanding them.

The current version of the Open Financial Exchange headers is version 1.0 (OFHEADER:100).

2.2.2 DATA

DATA specifies the content type, in this case OFXSGML.

You should add new values for a DATA tag only when you introduce an entirely new syntax. In the case of OFXSGML, a new syntax would have to be non-SGML compliant to warrant a new DATA value. It is possible that there will be more than one syntax in use at the same time to meet different needs.

2.2.3 VERSION

VERSION specifies the version number of the Document Type Definition (DTD) used for parsing. The current version of the DTDs is version 1.5 (VERSION:150).

The VERSION tag identifies syntactic changes. In the case of OFXSGML, this corresponds to the DTD. Purely for identification purposes, each change increments the minor number of the version tag. If you introduce an incompatible change so that an older DTD can not parse the file, the major number should change. See the general discussion of message sets and version control, later in this chapter.

> ***Note:*** *VERSION provides the version number of the DTD. The <OFX> block describes the version numbers of specific message sets.*

2.2.4 SECURITY

SECURITY defines the type of application-level security, if any, that is used for the <OFX> block. The values for SECURITY can be NONE or TYPE1.

For more information about security, refer to Chapter 4, “Open Financial Exchange Security.”

2.2.5 ENCODING and CHARSET

ENCODING defines the text encoding used for character data. The values for ENCODING can be USASCII or UTF-8.

CHARSET defines the character set used for character data.

For more information about ENCODING and CHARSET, refer to Chapter 5, “International Support.”

2.2.6 COMPRESSION

A future version of the specification will define compression.

2.2.7 OLDFILEUID and NEWFILEUID

NEWFILEUID uniquely identifies this request file. The NEWFILEUID, which clients must send with every request file and which servers must echo in the response, serves several purposes:

- Servers can use the NEWFILEUID to quickly identify duplicate request files.
- Clients and servers can use NEWFILEUID in conjunction with OLDFILEUID for file-based error recovery. For more information about using file-based error recovery or *lite synchronization*, see Chapter 6, “Data Synchronization.”
- Servers can use the NEWFILEUID to manage the session keys associated with Type 1 application-level security. For more information about security, refer to Chapter 4, “Open Financial Exchange Security.”

OLDFILEUID is used together with NEWFILEUID only when the client and server support file-based error recovery. OLDFILEUID identifies the last request and response that was received and processed by the client.

2.3 SGML Details

2.3.1 Compliance

SGML is the basis for Open Financial Exchange. A DTD formally defines the SGML wire format for Open Financial Exchange. However, Open Financial Exchange is not completely SGML-*compliant* because the specification allows unrecognized tags to be present. Clients and servers must skip over the unrecognized tags. That is, if a client or server does not recognize <XYZ>, it must ignore the tag and its enclosed data. A fully compliant SGML parser would not *validate* a document that contains tags that the DTD does not define.

Although SGML is the basis for the specification, and the specification is largely compliant with SGML, do not assume Open Financial Exchange supports any SGML features not documented in this specification.

2.3.2 Valid SGML Characters

Open Financial Exchange tags that require a value can be set to any sequence of SGML characters. To be valid, a value must contain at least one character that is not a blank character. In other words, a value cannot contain only white space.

2.3.2.1 Special Characters

For the purposes of Open Financial Exchange, a few characters must be handled as special characters. To represent a special character, use the corresponding escape sequence.

Character	*Escape sequence*
< (less than)	<
> (greater than)	>
& (ampersand)	&
‘ ‘ (space)	(added in OFX 1.5)

Note: *The space macro was not added until OFX 1.5. So when using message set version 1, there may be both clients and servers that can not process the   syntax.*

Note: *Space characters in the middle of a value do not require use of the special character macro. If a string value needs to contain space characters as the first or last characters, that’s when this macro is needed. However, the macro can also be used in the middle of a string value.*

For example, the string “AT&T” encodes “AT&T.”

Note: *Escape sequences are not required when these special characters are used in tag values that accept HTML-formatted strings (for instance, e-mail records). These tags accept SGML-marked section syntax that hides the HTML from the Open Financial Exchange parser. You must prefix the HTML-formatted strings with “<![CDATA [”and suffix them with “]]>.” Within these bounds, treat the above characters literally without an escape. See Chapter 9, “Customer to FI Communication,” for an example.*

2.3.3 Comments Not Supported

For explanatory purposes, the examples in this specification contain comments. However, Open Financial Exchange files *cannot* contain comments.

2.4 Open Financial Exchange SGML Structure

2.4.1 Overview

Open Financial Exchange hierarchically organizes request and response blocks:

```
Top Level <OFX>
   Message Set and Version <XXXMSGSVn>
      Synchronization Wrappers <YYYSYNCRQ>, <YYYSYNCRS>
         Transaction Wrappers <YYYTRNRQ>, <YYYTRNRS>
            Specific requests and responses
```

The following sections describe these levels.

2.4.2 Case Sensitivity

OFX requires upper case letters for element names and enumerated values. In the example below, <SEVERITY> is an element with an enumerated value and <MESSAGE> is an element with a value that is not enumerated.

```
<STATUS>
   <CODE> 2000
   <SEVERITY> ERROR
   <MESSAGE> General Error
</STATUS>
```

2.4.3 Top Level

An Open Financial Exchange request or response has the following top-level form:

Tag	*Description*
<**OFX**>	Opening tag
... Open Financial Exchange requests or responses ...	0 or more transaction requests and responses inside appropriate message set aggregates
</**OFX**>	Closing tag for the Open Financial Exchange record

This chapter specifies the order of requests and responses.

A single file MUST contain only one OFX block.

2.4.4 Messages

A message is the unit of work in Open Financial Exchange. It refers to a request and response pair, and the status codes associated with that response. For example, the message to download a bank statement consists of the request <STMTRQ> and the response <STMTRS>. In addition, with the exception of the signon message, each message includes a *transaction wrapper*. For requests, the transaction wrapper adds a transaction unique ID <TRNUID>. For responses, the transaction wrapper adds the same transaction unique ID <TRNUID>, plus a <STATUS> aggregate.

For messages subject to synchronization (see Chapter 6, “Data Synchronization”), a third layer of aggregates is also part of a message definition: a synchronization request and response. These add a token and, in some cases, other information.

Open Financial Exchange uses the following naming conventions where the *XXX* message includes:

- Basic request <*XXX***RQ**> and response <*XXX***RS**>
- Transaction wrapper <*XXX***TRNRQ**> and <*XXX***TRNRS**>
- If needed, synchronization wrapper <*XXX***SYNCRQ**> and <*XXX***SYNCRS**>

Note: *Some requests and responses share a transaction wrapper and synchronization wrapper. In these cases, the names of the transaction and synchronization wrappers do not follow the preceding naming conventions.*

2.4.5 Message Sets and Version Control

Message sets are collections of messages. Generally they form all or part of what a user would consider a *service*, something for which they might have signed up, such as "banking." Message sets are the basis of version control, routing, and security. They are also the basis for the required ordering in Open Financial Exchange files.

Within the Open Financial Exchange block, Open Financial Exchange organizes messages by message set. A message set can appear at most once within an Open Financial Exchange block. All messages from a message set must be from the same version of that message set.

2.4.5.1 Message Set Aggregates

For each message set of *XXX* and version *n*, there are two aggregates, one for requests <*XXX*MSGSRQV*n*>) and one for responses <*XXX*MSGSRSV*n*>. All of the messages from that message set must be enclosed in the appropriate message set aggregate. In the following example, the Open Financial Exchange block contains a signon request inside the signon message set, and two statement requests and a transfer request inside the bank message set.

```
<OFX>
   <SIGNONMSGSRQV1> <!-- Signon message set -->
      <SONRQ>                             <!-- Signon message -->
      ...
      </SONRQ>
   </SIGNONMSGSRQV1>

   <BANKMSGSRQV1>   <!-- Banking message set -->
      <STMTTRNRQ>   <!-- Statement request -->
      ...
      </STMTTRNRQ>
      <STMTTRNRQ>   <!-- Another stmt request -->
      ...
      </STMTTRNRQ>
      <INTRATRNRQ>                        <!-- Intrabank transfer request -->
      ...
      </INTRATRNRQ>
   </BANKMSGSRQV1>
</OFX>
```

2.4.5.2 Message Set Ordering

Message sets must appear in the following order:

- Signon
- Signup
- Banking
- Credit card statements
- Investment statements
- Interbank funds transfers
- Wire funds transfers
- Payments
- General e-mail
- Investment security list
- Biller Directory
- Bill Delivery
- FI Profile

The definition of each message set can further prescribe an order of its messages within that message set.

For ordering purposes, versions of a message set are grouped with the message set. For instance, Signon V2 would precede Banking V1, using the “natural” ordering above.

2.4.5.3 Message Set Version Numbers

Message sets have their own version numbers, which are distinct from the version numbers of the Open Financial Exchange headers and the Document Type Definition (DTD) files.

> ***Note:*** *The version numbers of the Open Financial Exchange headers and the Document Type Definition (DTD) files appear in the Open Financial Exchange headers, before the <OFX> data block. For more information about the Open Financial Exchange headers, see section 2.2. The current version number of the headers is OFXHEADER: 100. The current version number of the DTDs is VERSION: 150.*

The following table lists each message set, along with its aggregate name and current version number.

Message Set	*Message Set Aggregate*	*Version Number*
Signon	<SIGNONMSGSETV1>	1
Signon	<SIGNONMSGSETV2>	2
Signup	<SIGNUPMSGSETV1>	1
Signup	<SIGNUPMSGSETV2>	2
Banking	<BANKMSGSETV1>	1
Banking	<BANKMSGSETV2>	2
Credit Card Statements	<CREDITCARDMSGSETV1>	1
Credit Card Statements	<CREDITCARDMSGSETV2>	2
Investment Statements	<INVSTMTMSGSETV1>	1
Investment Statements	<INVSTMTMSGSETV2>	2
Interbank Funds Transfers	<INTERXFERMSGSETV1>	1
Interbank Funds Transfers	<INTERXFERMSGSETV2>	2
Wire Funds Transfers	<WIREXFERMSGSETV1>	1
Wire Funds Transfers	<WIREXFERMSGSETV2>	2
Payments	<BILLPAYMSGSETV1>	1
Payments	<BILLPAYMSGSETV2>	2
General e-mail	<EMAILMSGSETV1>	1
General e-mail	<EMAILMSGSETV2>	2
Investment security list	<SECLISTMSGSETV1>	1
Investment security list	<SECLISTMSGSETV2>	2
Biller directory	<PRESDIRMSGSETV1>	1
Bill delivery	<PRESDVLMSGSETV1>	1
FI Profile	<PROFMSGSETV1>	1
FI Profile	<PROFMSGSETV2>	2

> ***Note:*** *For each message set that it is supporting, a financial institution must indicate which version numbers of that message set it supports. The financial institution includes the message set version number in the <MSGSETCORE> aggregate of the FI profile. For more information about the FI profile, refer to Chapter 7, “FI Profile.”*

2.4.6 Transactions

Other than the signon message, each request is made as a transaction. Transactions contain a client-assigned globally-unique ID, optional client-supplied pass-back data, and the request aggregate. A transaction similarly wraps each response. The response transaction returns the client ID sent in the request, along with a status message, the pass-back data if present, and the response aggregate. This technique allows a client to track responses against requests.

The <STATUS> aggregate, defined in Chapter 3, "Common Aggregates, Elements, and Data Types," provides feedback on the processing of the request. If the <SEVERITY> of the status is ERROR, the server provides the transaction response without the nested response aggregate. Otherwise, the response must be complete even though some warning might have occurred.

Clients can send additional information in <CLTCOOKIE> that servers will return in the response. This allows clients that do not maintain state, and thus do not save <TRNUID>s, to cause some additional descriptive information to be present in the response. For example, a client might identify a request as relating to a user or a spouse.

Starting with the message sets new to OFX 1.5 (e.g., the V2 message sets and the bill presentation message sets), <CLTCOOKIE> must only be returned by the server in the initial response to the client (and any crash recovery from that response). The <CLTCOOKIE> should not be present in a sync response, except for those transactions whose requests were wrapped in the sync request. This restriction is new for message set version 2 of all the message sets.

In some countries, some banks may require that a customer-supplied authorization number be included to authenticate certain kinds of individual transactions such as payment requests. For those banks, the <TAN> element passes this information to servers.

Note that if a <CLTCOOKIE> is given to an OFX server in a request, the OFX server is required to return it. This return of the <CLTCOOKIE> will necessitate server-side storage of <CLTCOOKIE> data. In the case of an OFX client getting a <CLTCOOKIE> that it didn't send in a request, the default behavior is to ignore it.

A typical request is as follows:

Tag	*Description*
<*XXX*TRNRQ>	Transaction-request aggregate
<TRNUID>	Client-assigned globally-unique ID for this transaction, ***trnuid***
<CLTCOOKIE>	Data to be echoed in the transaction response, ***A-32***
<TAN>	Transaction authorization number; used in some countries with some types of transactions. The FI Profile defines messages that require a <TAN>, ***A-80***
Request aggregate	Aggregate for the request
</*XXX*TRNRQ>	

A typical response is as follows:

Tag	*Version*	*Description*
<*XXX*TRNRS>		Transaction-response aggregate
<TRNUID>		Client-assigned globally-unique ID for this transaction, ***trnuid***
<STATUS>		Status aggregate
</STATUS>		
<CLTCOOKIE>	V2 change	Client-provided data, **REQUIRED** if provided in request, ***A-32***
		See note above regarding usage inside sync response.
Response aggregate		Aggregate for the response
</*XXX*TRNRS>		

List of status code values for the <CODE> element of <STATUS>:

Value	*Meaning*
0	Success (INFO)
2000	General error (ERROR)
2022	Invalid TAN (ERROR)

2.5 The Signon Message Set

The Signon message set includes the signon message, USERPASS change message, and challenge message, which must appear in that order. The <SIGNONMSGSRQV1> and <SIGNONMSGSRSV1> (also V2) aggregates wrap the message.

2.5.1 Signon <SONRQ> <SONRS>

The signon record identifies and authenticates a user to an FI. It also includes information about the application making the request, because some services might be appropriate only for certain clients. Every Open Financial Exchange request contains exactly one <SONRQ>. Every response must contain exactly one <SONRS> record. Use of Open Financial Exchange presumes that FIs authenticate each customer and then give the customer access to one or more accounts or services. Authentication of a <SONRQ> is required, even when in Error Recovery. If passwords are specific to individual services or accounts, a separate Open Financial Exchange request must be made for each user ID or password required. This will not necessarily be in a manner visible to the user. Note that some situations, such as joint accounts or business accounts, will have multiple user IDs and multiple passwords that can access the same account.

FIs assign user IDs for the customer. Although the user ID may be the customer's social security number, the client must not make any assumptions about the syntax of the ID, add check-digits, or do similar processing. Servers must accept user IDs, with or without punctuation.

To improve server efficiency in handling a series of Open Financial Exchange request files sent over a short period of time, clients can request that a server return a <USERKEY> in the signon response. If the server provides a user key, clients will send the <USERKEY> instead of the user ID and password in subsequent sessions, until the <USERKEY> expires. This allows servers to authenticate subsequent requests more quickly. Servers must accept a <GENUSERKEY> element in a <SONRQ>. However, a server may decide <USERKEY> does not afford sufficient security and may optionally not return a <USERKEY> in the <SONRS>.

The client returns <SESSCOOKIE> if the server sent one in a previous <SONRS>. Servers can use the value of <SESSCOOKIE> to track client usage but cannot assume that all requests come from a single client, nor can they deny service if they did not expect the returned cookie. Use of a backup file, for example, would lead to an unexpected <SESSCOOKIE> value that nevertheless should not stop a user from connecting.

A client may use an anonymous form of <USERID> and <USERPASS> on those occasions when a server need not validate the <SONRQ>, i.e., <PROFRQ>, <ENROLLRQ>. The anonymous <USERID> or <USERPASS> value is left aligned and padded with 0 to a length of 32 characters: anonymous00000000000000000000000

Servers can request that a consumer change his or her password by returning status code 15000. Servers should keep in mind that only one status code can be returned. If the current signon response status should be 15500 (invalid ID or password), the request to change the password must wait until an otherwise successful signon is achieved.

If the server returns any signon error, it must respond to all other requests in the same <OFX> block with status code 15500. For example, if the server returns status code 15502 to the signon request, it must return status code 15500 to all other requests in the same <OFX> block. The server must return status code 15500 to all requests; it cannot simply ignore the requests.

An OFX 1.5 server has the option of allowing or disallowing "empty" signon transactions. In the context of signon, "empty" means a simple signon without any other transaction (a sync, statement download, etc). If the OFX 1.5 server does not support empty signon, it should return error 15506. If the OFX 1.5 server does support empty signon, it should process the signon and return the appropriate error or success code.

For OFX 1.0.2 servers, it is optional whether or not to support error 15506. Since additional error codes can be added to an OFX 1.0.2 server, it is suggested that logic to support error 15506 be added to OFX 1.0.2 servers that wish to disallow empty signon transactions. Older clients should remap unknown error codes to a general error.

2.5.1.1 Signon Request <SONRQ>

Unlike other requests, the signon request <SONRQ> does not appear within a transaction wrapper.

Tag	*Version*	*Description*
<SONRQ>		Signon-request aggregate
<**DTCLIENT**>		Date and time of the request from the client computer, ***datetime***
User identification. Either <USERID> and <USERPASS> or <USERKEY>, but not both.		
<USERID>		User identification string, ***A-32***
<USERPASS>		User password on server, ***A-171***
		Note: The effective size of USERPASS is A-32. However, if Type 1 security is used, then the actual field length is A-171.
<ONETIMEPASS>	V2 only	A special type of password that is used only once (in addition to USERPASS) to authenticate a single OFX session, ***A-32***
		Note: The client can send this tag only if the <PWTYPE> value returned by the server has the value of ONETIME or HWTOKEN (see section 7.2.2).
-or-		
<USERKEY>		Log in using previously authenticated context, ***A-64***
<GENUSERKEY>		Request server to return a USERKEY for future use, ***Boolean***
<LANGUAGE>		Requested language for text responses, ***language***
<COUNTRY>	V2 only	Specific country system used for the requests in this <OFX> block: 3-letter country code from ISO/DIS-3166.
		If this tag is not present, the country system is USA.
		The following countries are currently supported in OFX:
		COUNTRY Country Name
		BEL Belgium
		CAN Canada
		CHE Switzerland
		DEU Germany
		ESP Spain.
		FRA France
		GBR Great Britain
		ITA Italy
		NLD Netherlands
		USA United States of America
<FI>		Financial-Institution-identification aggregate
		Note: The client will determine out-of-band whether a FI aggregate should be used and if so, the appropriate values for it. If the FI aggregate is to be used, then the client should send it in every request, and the

Tag	Version	Description
		server should return it in every response.
</FI>		
<SESSCOOKIE>		Session cookie value received in previous <SONRS>, not sent if first login or if none sent by FI, ***A-1000***
<APPID>		ID of client application, ***A-5***
<APPVER>		Version of client application, (6.00 encoded as 0600), ***N-4***
</SONRQ>		

2.5.1.2 Signon Response <SONRS>

Unlike other responses, the signon response <SONRS> does not appear within a transaction wrapper.

Note: *A client should use <DTPROFUP> and <DTACCTUP> only when the service provider that originated SONRS is the same provider that is specified by <SPNAME> in the profile message set. A client can determine if the service provider is the same by comparing the value of <SPNAME> in the appropriate message set with the value for <SPNAME> in the profile message set.*

Tag	Version	Description
<SONRS>		Record-response aggregate
<STATUS>		Status aggregate, see list of possible code values
</STATUS>		
<DTSERVER>		Date and time of the server response, ***datetime***
<USERKEY>		Use user key instead of USERID and USERPASS for subsequent requests. TSKEYEXPIRE can limit lifetime. ***A-64***
<TSKEYEXPIRE>		Date and time that USERKEY expires, ***datetime***
<LANGUAGE>		Language used in text responses, ***language***
<COUNTRY>	V2 only	Specific country system used for the requests: 3-letter country code from ISO/DIS-3166. If this tag is not present, the country system is USA. ***Note: Where element behaviors are listed within this specification as dependent upon the value of <COUNTRY>, the reference is to this one in the <SONRS> aggregate. It is not to be confused with the element <COUNTRY> that appears elsewhere in OFX, associated with addresses.***
<DTPROFUP>		Date and time of last update to profile information for any service supported by this FI (see Chapter 7, "FI Profile"), ***datetime***
<DTACCTUP>		Date and time of last update to account information (see Chapter 8, "Activation & Account Information"), ***datetime***
<FI>		Financial-Institution-identification aggregate ***Note: The client will determine out-of-band whether an FI aggregate should be used and, if so, the appropriate values for it. If the FI aggregate is to be used, then the client should send it in every request, and the server should return it in every response.***
</FI>		
<SESSCOOKIE>		Session cookie that the client should return on the next <SONRQ>,***A-1000***
</SONRS>		

2.5.1.3 Status Codes

List of status code values for the <CODE> element of <STATUS>:

Value	Meaning

Value	Meaning
0	Success (INFO)
2000	General error (ERROR)
15000	Must change USERPASS (INFO)
15500	Signon invalid (ERROR); see section 2.5.1
15501	Customer account already in use (ERROR)
15502	USERPASS Lockout (ERROR)
15505	Country system not supported by server (ERROR)
15506	Empty signon transaction not supported (ERROR)

2.5.1.4 Financial Institution ID <FI>

Some service providers support multiple FIs, and assign each FI an ID. The signon allows clients to pass this information along, so that providers know to which FI the user is signing on.

Tag	Description
<FI>	FI-record aggregate
<ORG>	Organization defining this FI name space, ***A-32***
<FID>	Financial Institution ID (unique within <ORG>), ***A-32***
</FI>	

2.5.2 USERPASS Change <PINCHRQ> <PINCHRS>

The signon sends a request to change a customer password as a separate request. The transaction request <PINCHTRNRQ> aggregate contains <PINCHRQ>. Responses are placed inside transaction responses <PINCHTRNRS>. Password changes pose a special problem for error recovery. If the client does not receive a response, it does not know whether the password change was successful or not. Open Financial Exchange recommends that servers accept either the old password or the new password on the connection following the one containing a password change. The password used becomes the new password.

2.5.2.1 <PINCHRQ>

A USERPASS change request changes the customer's password for the specific realm associated with the messages contained in the OFX block. Based on the properties of an OFX profile, defined in Chapter 7, "FI Profile," a single OFX block contains instructions related to a single realm. The USERPASS change request thus changes the USERPASS for all message sets associated with one realm. For more information about signon realms, see section 7.2.2.

Tag	Description
<PINCHRQ>	USERPASS-change-request aggregate
<USERID>	User identification string. Often a social security number, but if so, does not include any check digits, ***A-32***
<NEWUSERPASS>	New user password, ***A-171***
	Note: The effective size of NEWUSERPASS is A-32. However, if Type 1 security is used, then the actual field length is A-171.
</PINCHRQ>	

2.5.2.2 <PINCHRS>

Tag	Description

Tag	Description
<PINCHRS>	USERPASS-change-response aggregate
<USERID>	User identification string. Often a social security number, but if so, does not include any check digits, ***A-32***
<DTCHANGED>	Date and time the password was changed, ***datetime***
</PINCHRS>	

2.5.2.3 <CHALLENGERQ> <CHALLENGERS>

A challenge request is the first step in Type 1 application-level security. Essentially, it asks for some random data from the server. The challenge response provides that server-generated random data and is the second step in Type 1 security.

The challenge message is part of the signon message set and is not subject to data synchronization.

A <CHALLENGERQ> is part of a <CHALLENGETRNRQ> transaction, a <CHALLENGERS> part of a <CHALLENGETRNRS>.

Tag	Description
<**CHALLENGERQ**>	Opening tag for the challenge request.
<USERID>	User identification string, ***A-32***
<FICERTID>	Optional server certificate ID. ***A-64***
</CHALLENGERQ>	Closing tag for challenge request.

Tag	Description
<**CHALLENGERS**>	Opening tag for the challenge response.
<USERID>	User identification string, ***A-32***
<NONCE>	Server-generated random data. ***A-16***
<FICERTID>	ID of server certificate used to encrypt. ***A-64***
</ **CHALLENGERS**>	Closing tag for challenge response.

When generating the <NONCE>, make sure the data is as unpredictable as possible. See RFC 1750 for recommendations.

The client includes <FICERTID> in the request if it already has the server's certificate. If it's included and matches the server's current certificate, the server may omit the actual certificate from the response.

The server includes <FICERTID> in the response to identify the certificate in a separate MIME part. Even if the certificate itself is not attached, <FICERTID> is still included in the response.

Status code values for the <CODE> element of <STATUS>:

Value	Meaning
0	Success (INFO)
2000	General error (ERROR)
15504	Could not provide random data (ERROR)

2.5.2.4 Status Codes

Value	Meaning
0	Success (INFO)
2000	General error (ERROR)
15503	Could not change USERPASS (ERROR)

2.5.3 Signon Message Set Profile Information

A server must include the signon message set <SIGNONMSGSET> as part of the <MSGSETLIST> aggregate in the FI profile, since every server must support signon requests.

The information that is part of the <MSGSETCORE> aggregate (for example, the URL and security level) is used only when no other message sets are used. Otherwise, the other message sets override the signon message set for the purposes of batching and routing. For example, if bill payments are sent to a URL that is different from the one used for signon, the client uses the URL specified in the bill payment message set <BILLPAYMSGSET>. For more information about how clients batch and route messages, refer to section 7.1.3.

Tag	*Description*
<SIGNONMSGSET>	Signon-message-set-profile-information aggregate
<SIGNONMSGSETV1>	Opening tag for V1 of the message set profile information
<MSGSETCORE>	Common message set information, defined in Chapter 7, “FI Profile”
</MSGSETCORE>	
</SIGNONMSGSETV1>	
<SIGNONMSGSETV2>	Opening tag for V2 of the message set profile information
<MSGSETCORE>	Common message set information, defined in Chapter 7, “FI Profile”
</MSGSETCORE>	
</SIGNONMSGSETV2>	
</SIGNONMSGSET>	

2.5.4 Examples

User requests a password change:

```
<PINCHTRNRQ>
   <TRNUID>888
   <PINCHRQ>
      <USERID>123456789
      <NEWUSERPASS>5321
   </PINCHRQ>
</PINCHTRNRQ>
```

The server responds with:

```
<PINCHTRNRS>
   <TRNUID>888
   <STATUS>
      <CODE>0
      <SEVERITY>INFO
   </STATUS>
   <PINCHRS>
      <USERID>123456789
   </PINCHRS>
</PINCHTRNRS>
```

2.6 External Data Support

Some data, such as binary data, cannot easily be sent within SGML. For these situations, the specification defines a tag that references some external data. The way that clients pick up the external data depends on the transport used. For the HTTP-based transport described in this document, servers can send the data in one of two ways:

- Send the same response, using multi-part MIME types to separate the response into the Open Financial Exchange file and one or more external data files
- Client can make a separate HTTP get against the supplied URL, if it really needs the data

For example, to retrieve a logo, a <GETMIMERS> might answer a <GETMIMERQ> as follows:

```
<GETMIMERS>
   <URL>https://www.fi.com/xxx/yyy/zzz.jpg
</GETMIMERS>
```

If the file includes the same response using multi-part MIME, clients must have the local file, zzz.jpg.

2.7 Extensions to Open Financial Exchange

An organization that provides a customized client and server that communicate by means of Open Financial Exchange might wish to add new requests and responses or even specific elements to existing requests and responses. To ensure that each organization can extend the specification without the risk of conflict, Open Financial Exchange defines a style of tag naming that lets each organization have its own name space.

Organizations can register a specific tag name prefix. (The specific procedure or organization to manage this registration will be detailed at a later time.) If an organization registers “ABC,” then they can safely add new tags named <ABC.SOMETHING> without

- Colliding with another party wishing to extend the specification
- Confusing a client or server that does not support the extension

The extensions are not considered proprietary. An organization is free to publish their extensions and encourage client and server implementers to support them.

All tag names that do not contain a period (.) are reserved for use in future versions of the Open Financial Exchange specification.

3. Common Aggregates, Elements, and Data Types

3.1 Common Aggregates

This section describes aggregates used in more than one service of Open Financial Exchange (for example, investments and payments).

3.1.1 Identification of Financial Institutions and Accounts

Open Financial Exchange does not provide a universal space for identifying financial institutions, accounts, or types of accounts. The way to identify an FI and an account at that FI depends on the service. For information about service-specific ID aggregates, see Chapters 11, 12, and 13 ("Banking," "Payments," and "Investments").

3.1.2 Format of User-Supplied Numbers

Clients will not attempt to strip dashes or other punctuation from user-supplied numbers, such as the <TAXID> in an enrollment request or the <*XXX*ACCTTO> in a service-addition request. Servers must be prepared to accept these numbers with or without punctuation.

3.1.3 Balance Records <BAL>

Several responses allow FIs to send an arbitrary set of balance information as part of a response, for example a bank statement download. FIs might want to send information on outstanding balances, payment dates, interest rates, and so forth. Balances can report the date the given balance reflects in <DTASOF>.

Tag	*Description*
<BAL>	Balance-response aggregate
<NAME>	Balance name, ***A-32***
<DESC>	Balance description, ***A-80***
<BALTYPE>	Balance type. DOLLAR = dollar (value formatted DDDD.cc) PERCENT = percentage (value formatted XXXX.YYYY) NUMBER = number (value formatted as is)
<VALUE>	Balance value. Interpretation depends on <BALTYPE> field, ***N-20***
<DTASOF>	Effective date of the given balance, ***datetime***
<CURRENCY>	If dollar formatting, can optionally include currency
</CURRENCY>	
</BAL>	

3.1.4 Error Reporting <STATUS>

To provide as much feedback as possible to clients and their users, Open Financial Exchange defines a <STATUS> aggregate. The most important element is the code that identifies the error. Each response defines the codes it uses. Codes 0 through 2999 have common meanings in all Open Financial Exchange transactions. Codes from 3000 and up have meanings specific to each transaction.

The last 10 error codes in each assigned range of 1000 is reserved for server-specific status codes. For example, of the general status codes, 2990-2999 are reserved for status codes defined by the server. Of the banking status codes, codes 10990-10999 are reserved for the server. If a client receives a server-specific status code of <SEVERITY> ERROR that it does not know, it will handle it as a general error 2000.

Tag	*Version*	*Description*
<STATUS>		Error-reporting aggregate.
<CODE>		Error code, ***N-6***
<SEVERITY>		Severity of the error: INFO = Informational only WARN = Some problem with the request occurred but a valid response still present ERROR = A problem severe enough that response could not be made
<MESSAGE>	V1 only	A textual explanation from the FI. Note that clients will generally have messages of their own for each error ID. Use this tag only to provide more details or for the general errors. ***A-255***
<MESSAGE2>	V2 only	A textual explanation from the FI. Note that clients will generally have messages of their own for each error ID. Use this tag only to provide more details or for the general errors. ***A-2000***
</STATUS>		

For general errors, the server can respond with one of the following <CODE> values. However, not all codes are possible in a specific context.

Code	*Meaning*
0	Success (INFO)
2000	General error (ERROR)
2021	Unsupported version (ERROR)
15500	Signon error (ERROR); see section 2.5.1

Note: *Clients will generally have error messages that are based on <CODE>. Therefore, do not use <MESSAGE> to replace that text. Use <MESSAGE> only to explain an error not well described by one of the defined codes, or to provide some additional information.*

3.2 Common Elements

This section defines elements used in several services of Open Financial Exchange. The format of the value is either alphanumeric (A-*n*) or numeric (N-*n*) with a maximum length *n*; or as a named type. Section 3.2.8 describes the named types.

3.2.1 Financial Institution Transaction ID <FITID>

Format: *A-255*

An FI assigns an <FITID> to uniquely identify a financial transaction that can appear in an account statement. Its primary purpose is to allow a client to detect duplicate responses. Open Financial Exchange intends <FITID> for use in statement download applications, where every transaction requires a unique ID; not just those that are client-originated or server-originated.

FITIDs must be unique within the scope of the requested transactions (that is, within an account) but need not be sequential or even increasing. Clients should be aware that FITIDs are not unique across FIs. If a client performs the same type of request within the same scope at two different FIs, clients will need to use FI + account + <FITID> as a unique key in a client database.

> ***Note:*** *Although the specification allows FITIDs of up to 255 alphanumeric characters, client performance may significantly improve if servers use fewer characters. It is recommended that servers use 32 characters or fewer.*

Usage: Bank statement download, investment statement download

3.2.2 Server-Assigned ID <SRVRTID>, <SRVRTID2>

Format: *A-10 for <SRVRTID>, used in V1 message sets; A-36 for <SRVRTID2>, used in V2 message sets*

A <SRVRTID> is a server-assigned ID for an object that is stored on the server. It should remain constant throughout the lifetime of the object on the server. The client will consider the SRVRTID as its "receipt" or confirmation and will use this ID in any subsequent requests to change, delete, or inquire about this object.

Where the context allows, a server can use the same *value* for a given server object for both <SRVRTID> and <FITID>, but the client will not know this. <SRVRTID>s must be unique only within the scope of the requests and responses they apply to, such as an account number. Like <FITID>, a <SRVRTID> is not unique across FIs and clients might need to use FI + <SRVRTID> if a client requires a unique key.

Usage: Payments, Banking

3.2.3 Client-Assigned Transaction UID <TRNUID>

Format: *A-36*

Open Financial Exchange uses <TRNUID>s to identify transactions within transaction wrappers (<*XXX*TRNRQ>, </*XXX*TRNRQ>).

In most cases, clients originate <TRNUID>s. When a client originates a <TRNUID>, the value of the <TRNUID> is always set to a unique identifier. Clients expect the server to return the same <TRNUID> in the corresponding response and can use this <TRNUID> to match up requests and responses. Servers can use <TRNUID>s to reject duplicate requests. Because multiple clients might be generating requests to the same server, transaction IDs must be unique across clients. Thus, <TRNUID> must be a globally unique ID.

In some cases, servers can originate a transaction that was not specifically requested by a client. For instance, a client might set up a recurring payment model. Although the client originates the payment model, the server originates the individual payments. Whenever the server originates a transaction, the value of the <TRNUID> must be set to zero.

The Open Software Foundation Distributed Computing Environment standards specify a 36-character hexadecimal encoding of a 128-bit number and an algorithm to generate it. Clients are free to use their own algorithm, to use smaller <TRNUID>s, or to relax the uniqueness requirements. However, it is **RECOMMENDED** that clients allow for the full 36 characters in responses to work better with other clients.

Usage: All services

3.2.4 Token <TOKEN>, <TOKEN2>

Format: *A-10 for <TOKEN>, used in V1 message sets; A-36 for <TOKEN2>, used in V2 message sets*

Open Financial Exchange uses <TOKEN> as part of data synchronization requests to identify the point in history that the client has already received data, and in responses to identify the server's current end of history. See Chapter 6, "Data Synchronization," for more information.

<TOKEN> is unique within an FI and the scope of the synchronization request. For example, if the synchronization request includes an account ID, the <TOKEN> needs to be unique only within an account. Servers are free to use a <TOKEN> that is unique across the entire FI. Clients must save separate <TOKEN>s for each account, FI, and type of synchronization request.

Usage: All synchronization requests and responses

3.2.5 Transaction Amount <TRNAMT>

Format: *Amount*

Open Financial Exchange uses <TRNAMT> in any request or response that reports the total amount of an individual transaction.

Usage: Bank statement download, investment statement download, payments

3.2.6 Memo <MEMO>, <MEMO2>

Format: *A-255 for <MEMO>, used in V1 message sets; A-390 for <MEMO2>, used in V2 message sets*

A <MEMO> provides additional information about a transaction.

Usage: Bank statement download, investment statement download, payments, transfers

3.2.7 Date Start and Date End <DTSTART> <DTEND>

Format: *Datetime*

Clients use these tags in requests to indicate the range of response that is desired. Servers use these tags in responses to let clients know what the FI was able to produce.

In requests, the following rules apply:

- If <DTSTART> is absent, the client is requesting all available history (up to the <DTEND>, if specified). Otherwise, it indicates the *inclusive* date and time in history where the client expects servers to start sending information.
- If <DTEND> is absent, the client is requesting all available history (starting from <DTSTART>, if specified). Otherwise, it indicates the *exclusive* date and time in history where the client expects servers to stop sending information.

In responses, the following rules apply:

- <DTSTART> is the date and time where the server began *looking* for information, not necessarily the date of the earliest returned information. If the response <DTSTART> is later than the requested <DTSTART>, clients can infer that the user has not signed on frequently enough to ensure that the client has retrieved all information. If the user has been calling frequently enough, <DTSTART> in the response will match <DTSTART> in the request.
- <DTEND> is the date and time that, if used by the client as the next requested <DTSTART>, it would pick up exactly where the current response left off. It is the *exclusive* date and time in history where the server stopped *looking* for information, based on the request <DTEND> rules.

In all cases, servers are **REQUIRED** to use a "system add datetime" as the basis for deciding which details match the requested date range. For example, if an FI posts a transaction dated Jan 3 to a user's account on Jan 5, and a client connects on Jan 4 and again on Jan 6, the server is **REQUIRED** to return that Jan 3-dated transaction when the client calls on Jan 6.

Usage: Bank statement download, investment statement download

3.2.8 Common Data Types

3.2.8.1 Dates, Times, and Time Zones

There is one format for representing dates, times, and time zones. The complete form is:

YYYYMMDDHHMMSS.XXX [*gmt offset:tz name*]

3.2.8.2 Date and Datetime

Tags specified as type *date or datetime* and generally starting with the letters "DT" accept a fully formatted date-time-timezone string. For example, "19961005132200.124[-5:EST]" represents October 5, 1996, at 1:22 and 124 milliseconds p.m., in Eastern Standard Time. This is the same as 6:22 p.m. Greenwich Mean Time (GMT).

Date and *datetime* also accept values with fields omitted from the right. They assume the following defaults if a field is missing:

Specified date or datetime	*Assumed defaults*
YYYYMMDD	12:00 AM (the start of the day), GMT
YYYYMMDDHHMMSS	GMT
YYYYMMDDHHMMSS.XXX	GMT

Note that times zones are specified by an offset and optionally, a time zone name. The offset defines the time zone. Valid offset values are in the range from –12 to +12 for whole number offsets. Formatting is +12.00 to -12.00 for fractional offsets, plus sign may be omitted.

Take care when specifying an ending date without a time. If the last transaction returned for a bank statement download was Jan 5 1996 10:46 a.m. and if the <DTEND> was given as just Jan 5, the transactions on Jan 5 would be resent. If results are available only daily, then just using dates and not times will work correctly.

> ***Note:*** *Open Financial Exchange does not require servers or clients to use the full precision specified. However, they are* ***REQUIRED*** *to accept any of these forms without complaint.*

Some services extend the general notion of a *date* by adding special values, such as “TODAY.” These special values are called “smart dates.” Specific requests indicate when to use these extra values, and list the tag as having a special data type.

3.2.8.3 Time

Tags specified as type *time* and generally ending with the letters “TM” accept times in the following format:

HHMMSS.XXX[*gmt offset*:*tz name*]

The milliseconds and time zone are still optional, and default to GMT.

3.2.8.4 Time Zone Issues

Several issues arise when a customer and FI are not in the same time zone, or when a customer moves a computer into new time zones. In addition, it is generally unsafe to assume that computer users have correctly set their time or time zone.

Although most transactions are not sensitive to the exact time, they often are sensitive to the date. In some cases, time zone errors lead to actions occurring on a different date than intended by the customer. For this reason, servers should always use a complete local time plus GMT offset in any datetime values in a response. If a customer’s request is for 5 p.m. EST, and a server in Europe responds with 1 a.m. MET the next day, a smart client can choose to warn the customer about the date shift.

Clients that maintain local state, especially of long-lived server objects, should be careful how they store datetime values. If a customer initiates a repeating transaction for 5 p.m. EST, then moves to a new time zone, the customer might have intended that the transaction remain 5 p.m. in the new local time, requiring a change request to be sent to the server. If, however, the customer intended it to remain fixed in server time, this would require a change in the local time stored in the client.

Client software that doesn’t know the current local time zone for the user, or client proxies that don’t know the current local time zone of their end users, should maintain and display the datetime value in the time zone indicated by the originator of the value and explicitly marked with that time zone. As an example, consider <DTPMTDUE> in section 11.5.4.2. If the biller gave a due date of 23:59pm EST on Dec 29, 1997, this is best displayed as 23:59pm EST rather than rendered in local time if there is any doubt at all as to the current local time zone of the end user looking at the due date.

3.2.9 Amounts, Prices, and Quantities

3.2.9.1 Basic Format

Format: *A-32*

This section describes the format of numerical values used for amounts, prices, and quantities. In all cases, a numerical value that does not contain a decimal point has an implied decimal point at the end of the value. For example, a numerical value of “550” is equivalent to “550.” Trailing and leading spaces should be stripped. Number format uses a leading sign. Negative number format uses a minus sign (-). Positive number format uses a plus sign (+). The plus sign is implied for all amounts and can be omitted.

The following types are defined to have a maximum of 32 alphanumeric characters, including digits and punctuation. However, clients and servers may have specific limits for the maximum number of digits to the left or right of a decimal point. If a server cannot support a client request due to the size or precision of a number, the server should return status code 2012.

Amount: Amounts that do not represent whole numbers (for example, 540.32), must include a decimal point or comma to indicate the start of the fractional amount. Amounts should not include any punctuation separating thousands, millions, and so forth. The maximum value accepted depends on the client.

Quantity: Use decimal notation.

Integer: A whole number.

Unitprice: Use decimal notation. Unless specifically noted, prices should always be positive.

Rate: Use decimal notation, with the rate specified out of 100%. For example, 5.2 is 5.2%.

Some services define special values, such as INFLATION, which you can use instead of a designated value. Open Financial Exchange refers to these as "smart types," and identifies them in the specification.

3.2.9.2 Positive and Negative Signs

Unless otherwise noted in the specification, Open Financial Exchange always signs amounts and quantities from the perspective of the customer. Some typically negative amounts:

- Investment buy amount, investment sell quantity
- Bank statement debit amounts, checks, fees
- Credit card purchases
- Margin balance (unless the FI owes the client money)

Some typically positive amounts:

- Investment sell amount, investment buy quantity
- Bank statement credits
- Credit card payments
- Ledger balance (unless the account is overdrawn)

3.2.10 Language

Language identifies the human-readable language used for such things as status messages and e-mail. *Language* is specified as a three-letter code based on ISO-639.

3.2.11 Other Basic Data Types

Boolean: Y = yes or true, N = no or false.

currsymbol: A three-letter code that identifies the currency used for a request or response. The currency codes are based on ISO-4217. For more information about currencies, refer to section 5.2.

URL: String form of a World Wide Web Uniform Resource Location. It should be fully qualified including protocol, host, and path. *A-255.*

URL2: URL2 was added in OFX 1.5 to handle URLs that might contain longer server names, security tokens and context data. String form of a World Wide Web Uniform Resource Location. It should be fully qualified including protocol, host, and path. *A-1024.*

4. Open Financial Exchange Security

Open Financial Exchange (OFX) provides several options for ensuring the security of customer transactions. This chapter describes the OFX security framework, security goals, types of security, and financial institution (FI) responsibilities.

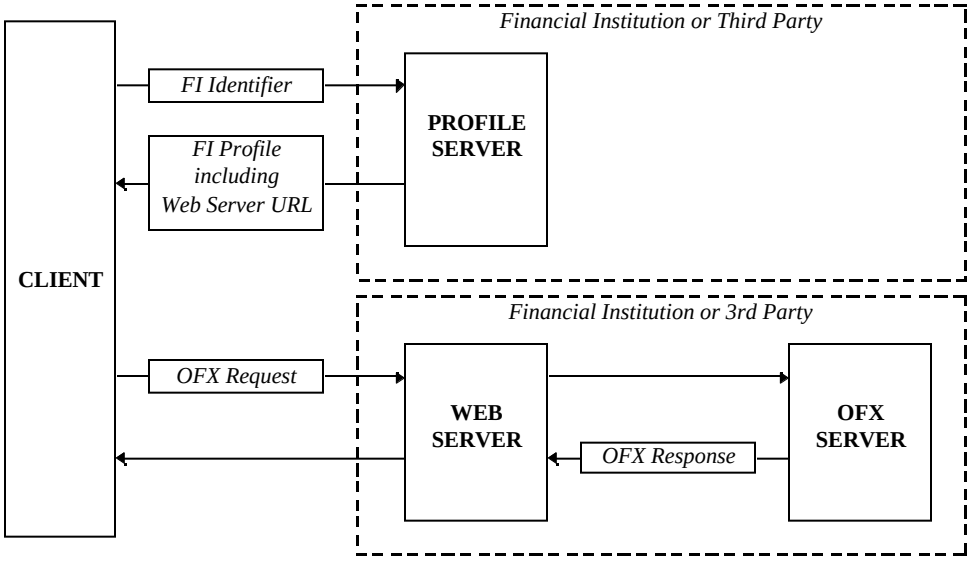
4.1 Security Concepts in OFX

4.1.1 Architecture

Open Financial Exchange security applies to the communication paths between a client and the profile server, a client and the Web server, and, when the OFX server is separate from the Web server, a client and the OFX server. The diagram below illustrates the initial order in which these communications occur, assuming that the client already has the URL for the FI profile server.

The bootstrap process for a client is:

- From the FI Profile Server, the client gets the URL of the FI Web server, so that it can retrieve a particular message set.
- The client sends an OFX request to the FI Web Server URL, from which it is forwarded to the OFX Server.
- The OFX Server sends back a response to the client via the Web Server.



4.1.2 Security Goals

The main goals of Open Financial Exchange security are:

- **Privacy**: Only the intended recipient can read a message. *Encryption* is a technique often used to ensure privacy.
- **Authentication**: The recipient of a message can verify the identity of the sender. In OFX, *passwords* allow an FI to authenticate a client, and *certificates* allow a client to authenticate a server.
- **Integrity**: A message cannot be altered after it is created A cryptographic *hash* is often used to assist integrity verification.

Open Financial Exchange specifies the minimum security required for Internet transactions and provides several security options, based on existing standards. Through its choice of security techniques and related options, an FI can achieve privacy, authentication, and integrity with varying degrees of assurance. For example, there are many kinds of encryption algorithms, most of which can be strengthened or weakened by changing the key size.

4.1.3 Security Standards

Several standards underlie Type 1 security:

- Certificates (X.509 v3) are used to identify and authenticate servers, and to convey their public keys.
- PKCS #1 block type 2 is the encryption format specified by the recipe (See Section 4.2.2.4.3).
- RSA is the encryption algorithm.

4.1.3.1 Certificates and Certification Authorities

A certificate is a digitally signed document that binds a public key to an identity. It contains a public key that identifies information such as the name of the person or organization to whom the key belongs, an expiration date, a unique serial number, and additional descriptive information.

A certificate is useful for authentication because it is signed by a trusted third-party. The assures the verifier that the certificate has not been changed since it was signed. The entity which signs certificates is called a *certification authority*, or CA. A CA acts somewhat like a notary public: the reader of a document stamped by a notary public knows that the notary has checked the identity of the person who originated the document. By digitally signing someone's identity and public key, the CA affirms that the two go together.

If the client and server do not share a common CA, the client cannot validate the server's certificate. For this reason, Open Financial Exchange specifies a number of trusted CAs that all clients must accept and all servers must use.

Certificates are used in Type 1 security, as well as channel-level security through SSL. The format for these is defined by X.509 version 3. For more information, refer to ITU-T Rec. X.509, ISO/IEC 9594-8.

4.1.3.2 PKCS #1

The acronym, PKCS, stands for "Public Key Cryptography Standards," a set of standards developed by a consortium and hosted by RSA. PKCS #1 is the RSA Encryption Standard, the rules for using RSA public key encryption. For the complete syntax of the PKCS #1 standard, refer to "Public-Key Cryptography Standards (PKCS)" published by RSA Data Security, Inc. at http://www.rsa.com/.

4.1.4 FI Responsibilities

Open Financial Exchange is designed with the understanding that there must be a security policy in place at each supporting financial institution. That policy must clearly delineate how customer data is secured, and how transactions are managed such that all parties to the transaction are protected according to accepted and recognized best common practices.

The decision regarding which users may perform a given operation on a given account must be determined by the financial institution. For example, is the specified user authorized to perform a transfer from the specified account? The financial institution must also determine whether the user has exceeded allowed limits on withdrawals, whether the activity on this account is unusual given past history, and other context-sensitive issues.

Although Open Financial Exchange provides many security options, an FI must support a minimal level of security. To ensure the proper security configuration, an FI must follow the steps outlined below.

1. Obtain one certificate for the profile server. This certificate must be rooted in one of the approved Certification Authorities (CAs). Establish appropriate safeguards for this certificate and its private key.
2. Obtain a certificate, rooted in an acceptable CA, for each OFX server, whether it is operated by the FI or by a third party.

3. Decide whether to use Type 1 application-level security for any message sets. For each message set to be secured by Type 1, obtain a certificate.

Type 1 security can be used on any message set, except for the Profile message set.

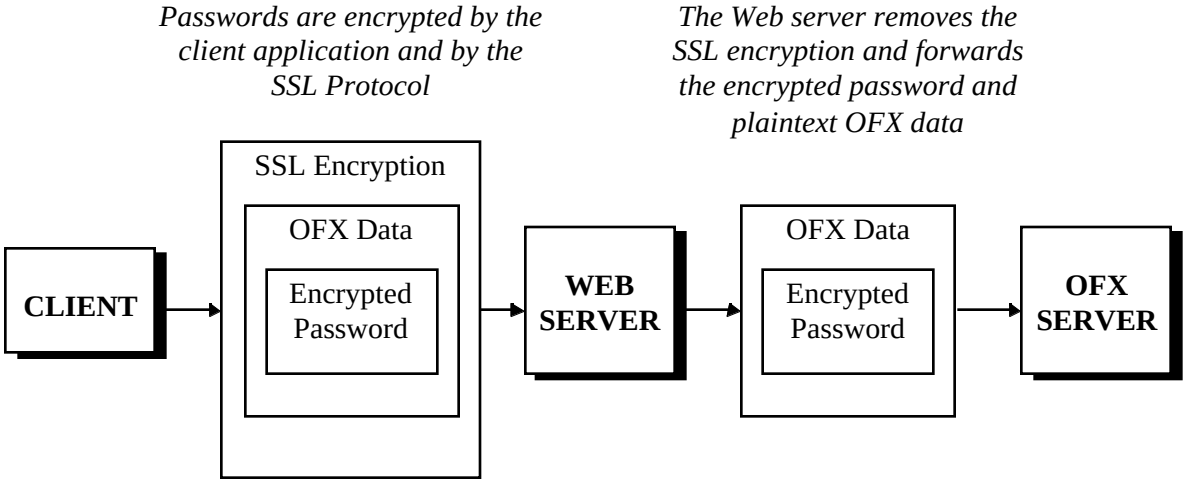
There are a number of other security issues beyond Open Financial Exchange proper, especially those relating to the Internet and network engineering. These issues are beyond the scope of this document. FIs are advised to conduct a complete security review of all servers associated with Open Financial Exchange.

4.1.5 Security Levels: Channel vs. Application

With Open Financial Exchange, security can be applied at two different levels in the message exchange process.

- **Channel level:** Generally transparent to a client or server, channel-level security is built into the communication process, protecting messages between two ends of the "pipe." To secure messages during HTTP transport, client and server applications use the Secure Sockets Layer (SSL) protocol. SSL transparently protects messages exchanged between the client and the destination Web server. SSL authenticates the destination Web server using the Web server's certificate. Additionally, it provides privacy via encryption, and SSL-record integrity, i.e. the block of data sent in each transmission cannot be altered without detection.
- **Application level:** Transparent to and independent of the transport process, application-level security protects the user password sent from the client application all the way to the server application that handles the Open Financial Exchange messages. The server application typically resides beyond the destination Web server, secured behind an Internet firewall. Application-level security requires channel-level security.

The following diagram illustrates how channel-level and application-level security relate. The diagram shows the path of a request from the client to the server when application-level encryption is used.



Channel-level security is sufficient for most message sets, provided that the network architecture at the destination is adequately secure; however, application-level password encryption can allow a more flexible back-end architecture with a high level of security.

4.2 Security Implementation in OFX

4.2.1 Channel-Level Security

4.2.1.1 Specification in FI Profile

For each message set listed in the FI profile response, the <MSGSETCORE> aggregate describes the channel-level security required for that message set.

The <TRANSPSEC> element defines whether or not channel-level security is required. It can have one of the following values:

Tag	Description
N	Do not use any channel-level security
Y	Use channel-level security

All currently defined message sets require channel-level security.

4.2.1.2 SSL Protocol

Secure Sockets Layer (SSL) is a cryptographic protocol commonly used for channel-level security on the Internet. Central to the security of SSL is the *server certificate*. This certificate assures clients that the server is who it claims to be. It contains the public key of the server, which the client uses to encrypt the session keys it generates as part of each connection.

All of this function is available without significant software development on either the client or server side; however, the client and server must be configured to use appropriate encryption algorithms (CipherSuites). In addition, clients and servers must share a trusted root certificate, or the client will not be able to validate the server's certificate.

> ***Note:*** *Although SSL supports client-side certificates to allow a server to authenticate a client, Open Financial Exchange does not require them at this time. To identify and authenticate a customer, servers should use the information provided in the signon request <SONRQ>.*

Setting the <TRANSPSEC> element to Y means that the client must use SSL v3 or higher.

4.2.1.3 Trusted Certificate Authorities

Both channel-level and application-level security rely on clients and servers having at least one trusted certification authority (CA) in common. To ensure that clients can test the validity of a certificate, servers must have their certificates signed by an approved OFX CA[1]. Clients are assumed to have access to this trusted CA.

4.2.1.4 CipherSuites

The following SSL CipherSuites are approved for use with OFX:

- SSL_RSA_WITH_RC4_128_SHA
- SSL_RSA_WITH_IDEA_CBC_SHA
- SSL_RSA_WITH_DES_CBC_SHA
- SSL_RSA_WITH_3DES_EDE_CBC_SHA
- SSL_DH_DSS_WITH_DES_CBC_SHA
- SSL_DH_DSS_WITH_3DES_EDE_CBC_SHA
- SSL_DH_RSA_WITH_DES_CBC_SHA
- SSL_DH_RSA_WITH_3DES_EDE_CBC_SHA
- SSL_DHE_DSS_WITH_DES_CBC_SHA
- SSL_DHE_DSS_WITH_3DES_EDE_CBC_SHA
- SSL_DHE_RSA_WITH_DES_CBC_SHA
- SSL_DHE_RSA_WITH_3DES_EDE_CBC_SHA

Other CipherSuites are not approved.

4.2.1.5 Key Size

Signing keys must be either RSA with a minimum 1024-bit modulus, or DSS with a 1024-bit modulus.

Server RSA keys and Diffie-Hellman keys must both have a minimum 1024-bit modulus. The Diffie-Hellman base must be primitive.

4.2.2 Application-Level Security

4.2.2.1 Specification in FI Profile

For each message set listed in the FI profile response, the <MSGSETCORE> aggregate describes the security required for that message set.

The <OFXSEC> element defines the type of application-level security required for the message set. <OFXSEC> can have one of the following values, which also are used in the SECURITY element of the OFX headers:

Tag	*Description*
NONE	Do not use any application-level security
TYPE1	Use Type 1 application-level security

[1] Approved OFX CAs will be identified at a later date.

Application-level security requires channel-level security.

4.2.2.2 Type 1 Protocol Overview

The goal of the Type 1 protocol is to protect the user password all the way to the destination OFX server. In the absence of client certificates, this password is the primary vehicle for client authentication and is therefore worthy of special consideration.

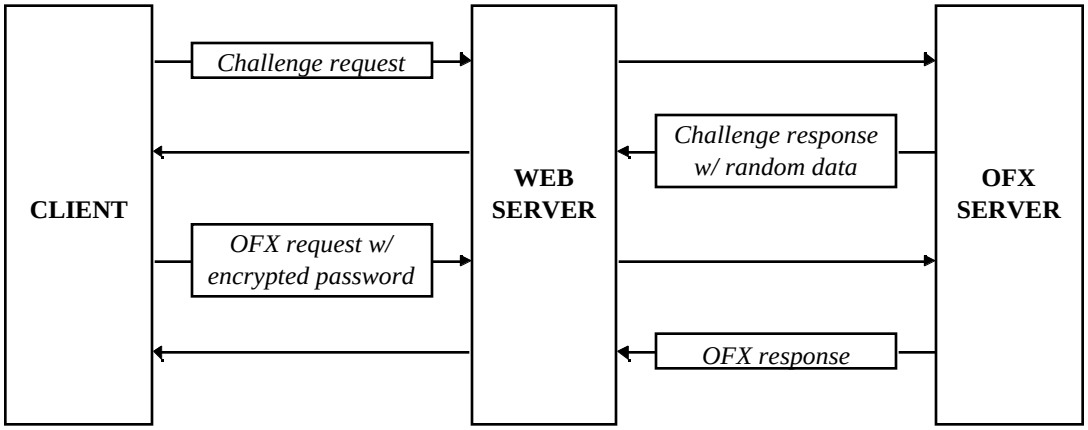
Type 1 requires channel-level security, *i.e.* SSL. Though the password is well protected by SSL alone in the client to Web server connection, the server-side network architecture may render the password less secure while it is in transit between the Web and OFX servers. With Type 1, the user password is not decrypted until the request reaches the OFX server.

Type 1 applies only to the request part of a message; the server response is unaffected.

A simple approach would be to deliver the server's Type 1 certificate in the profile and use it to encrypt the password, but that would permit a *replay attack.* An attacker could capture a transaction, including encrypted password, and replay it to the server. It wouldn't matter that the password remained unknown.

To prevent the *replay attack*, the server introduces some random data to the process, data which is unpredictably different for each transmission. The client asks for the random data with a challenge request. The server sends it, along with its Type 1 certificate, in the challenge response. The client then uses that random data in the encryption process, thereby assuring the server that the client response is associated with this and only this interaction.

The following diagram illustrates:



4.2.2.3 Type 1 Protocol Notation

In this section, the expression, $C = E_A(M)$, means that plain text M is encrypted either symmetrically or asymmetrically with key A into ciphertext C. The expression, $M = D_A(C)$ signifies the inverse operation (decryption), in which ciphertext C is decrypted into plain text M using key A. If C was encrypted asymmetrically, then A in the latter case is understood to be the private component of the key. The expression, A || B, indicates that B is concatenated to A.

4.2.2.4 Type 1 Protocol Implementation

Type 1 application-level security provides additional password secrecy. These are the steps for conducting a Type 1 transaction (unless otherwise noted, the term "Server" in this section refers to the Financial Institution Server):

1. Client obtains the Server's profile from the Profile Server (see Chapter 7, "FI Profile")
2. Client establishes an SSL connection with the Server (see Section 4.2.1)
3. Client sends <CHALLENGERQ> to Server (see Section 4.2.2.4.1)

4. Server sends <CHALLENGERS> which contains a nonce and the Server's Type 1 certificate (see Section 4.2.2.4.2)
5. Client builds a transaction request and sends it to the Server (see Section 4.2.2.4.3)
6. Server parses the request, verifying the user password, and either rejects or processes the transaction (see Section 4.2.2.4.4)

The following table lists data elements used in the Type 1 protocol:

Field	Type	Description
BT	octet, length 1	Block Type byte. BT = 0x02
CT1	octet string, length 128	Ciphertext: the PKCS #1 RSA encryption of EB with KS. CT1 = E_{KS}(EB)
CT2	printable ASCII, length 171	Encoded Ciphertext: the RADIX-64 encoding of CT1 (see RFC 1113, §4.3.2.4 and §4.3.2.5). CT2 = RADIX64(CT1)
D	octet string, length 68	Data: the user data to be encrypted. D = NC || P || T
EB	octet string, length 128	Encryption Block: the formatted plain text block, ready for encryption. EB = 0x00 || BT || PS || 0x00 || D
KS	RSA key, modulus length 1,024 bits	Server's Type 1 RSA key
NC	octet string, length 16	Client Nonce: string of random octets generated by the Client
NS	octet string, length 16	Server Nonce: string of random octets generated by the Server
P	printable ASCII, null-padded, length 32	Password: shared by the Client and Financial Institution, null-padded on the right
PS	octet string, length 57	Padding String: each octet is pseudo-random and non-zero
T	octet string, length 20	Authentication Token. T = SHA1(NS || P || NC)

```
struct {
   unsigned char nc[16];
   unsigned char p[32];
   unsigned char t[20];
} D;
struct {
   unsigned char null1 = 0x00;
   unsigned char bt = 0x02;
   unsigned char ps[57];
   unsigned char null2 = 0x00;
   struct D d;
} EB;
```

4.2.2.4.1 Challenge request

Client sends a <CHALLENGERQ> to the Server.

4.2.2.4.2 Challenge response

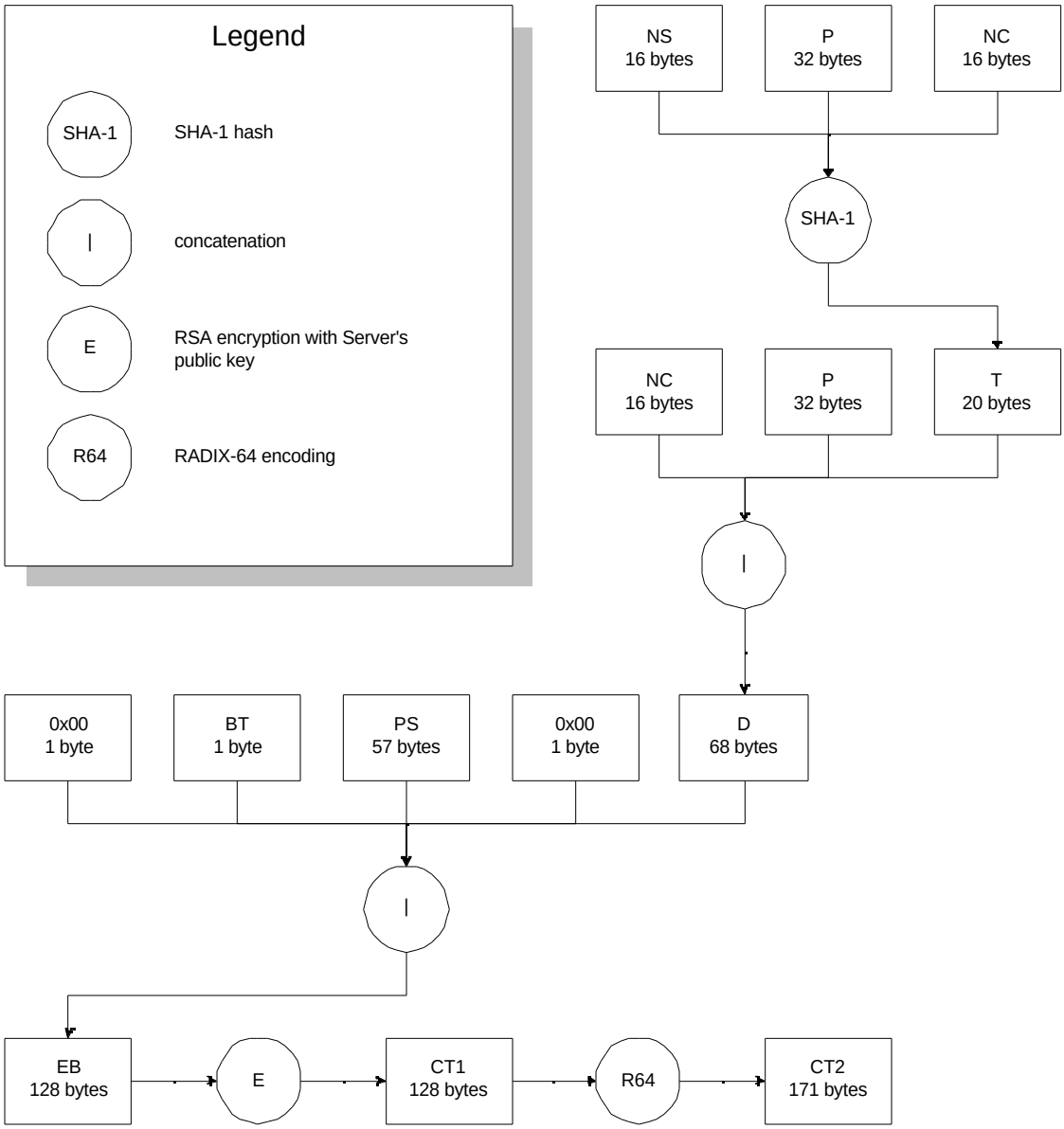
Server sends a <CHALLENGERS> to the client. This response contains the Server's Type 1 certificate and NS.

4.2.2.4.3 Building the OFX Request

1. Client generates 16 random octets and places them in NC (see RFC 1750 for recommendations on entropy generation)
2. Client obtains the User's password (P)
3. Client computes T = SHA1(NS || P || NC)
4. Client generates 57 pseudo-random, non-zero octets and places them in PS (NC may be used to seed the pseudo-random number generator)
5. Client sets D = NC || P || T
6. Client sets EB = 0x00 || BT || PS || 0x00 || D
7. Client RSA-encrypts EB using the Server's Type 1 public key (obtained from the Server's Type 1 certificate): CT1 = E_{KS}(EB) (see PKCS #1, §§8.2-8.4)
8. Client encodes the ciphertext for transport: CT2 = RADIX64(CT1). See RFC 1113, §4.3.2.4 and §4.3.2.5. This is a standard encoding method supported by RSA's Bsafe library and others.
9. Client constructs the body of its OFX request
10. Client copies CT2 to the <USERPASS> field of the OFX <SONRQ>
11. Client sends the complete OFX request to the Server

In <PINCHRQ>, the steps are identical, except that in step 2, P is set to <NEWUSERPASS> and in step 10, CT2 is copied to the <NEWUSERPASS> field of the <PINCHRQ>.

The diagram below illustrates the creation of CT2.



4.2.2.4.4 Parsing the OFX Request

1. Server reads the <SECURITY> field in the OFX header to ascertain whether Type 1 processing should be used on this message. If Type 1 is not used, skip to step 7
2. Server extracts CT2 from the <USERPASS> field of the OFX <SONRQ> and removes the encoding to obtain CT1 (see RFC 1113, §4.3.2.4 and §4.3.2.5)
3. Server decrypts CT1 to obtain EB: EB = D_{KS}(CT1) (see PKCS #1, §9)
4. Server extracts D from EB, then extracts NC, P, and T from D

5. Server looks up the Client's password in its database, and computes SHA1(NS || P || NC). If the result does not match T, Server terminates the session and reports the error to the client
6. Server processes the request and returns confirmation to the Client

In <PINCHRQ>, the steps are identical except that in step 2, CT2 is obtained from the <NEWUSERPASS> field of the OFX <PINCHRQ>.and in step 5, the server does not look up the extracted new password in a database.

5. International Support

5.1 Language and Encoding

Most of the content in Open Financial Exchange is language-neutral. However, some error messages, balance descriptions, and similar tags contain text meant to appear to the financial institution customers. There are also cases, such as e-mail records, where customers need to send text in other languages. To support worldwide languages, Open Financial Exchange must identify the basic text encoding, character set, and language.

The Open Financial Exchange headers specify the encoding and character set, as described in Chapter 2, "Structure." Current encoding values are USASCII and UTF-8. For USASCII, character set values are code pages. UNICODE ignores the character set *per se* although it still requires the syntax. UTF-8 is a form of UNICODE defined in International ISO/IEC 10646-1, amendment 2. Servers must respond with the encoding and character set requested by the client.

Clients identify the language in the signon request. Open Financial Exchange specifies languages by three-letter codes as defined in ISO-639. Servers report their supported languages in the profile (see Chapter 7, "FI Profile"). If a server cannot support the language requested by the client, it must return an error and not process the rest of the transactions.

5.2 Currency <CURDEF> <CURRENCY> <ORIGCURRENCY>

In each transaction involving amounts, responses include a default currency identification, <CURDEF>. The values are based on the ISO-4217 three-letter currency identifiers.

Within each transaction, specific parts of the response might need to report a different currency. Where appropriate, aggregates include an optional <CURRENCY> aggregate. The scope of a <CURRENCY> aggregate is everything within the same aggregate that the <CURRENCY> aggregate appears in, including nested aggregates, unless overridden by a nested <CURRENCY> aggregate. For example, specifying a <CURRENCY> aggregate in an investment statement detail means that the unit price, transaction total, commission, and all other amounts are in terms of the given currency, not the default currency.

Note that there is no way for two or more individual elements that represent amounts—and are directly part of the same aggregate—to have different currencies. For example, there is no way in a statement download to have a different currency for the <LEDGERBAL> and the <AVAILBAL>, because they are both directly members of <STMTRS>. In most cases, you can use the optional <BAL> aggregates to overcome this limitation, since <BAL> aggregates accept individual <CURRENCY> aggregates.

The default currency for a request is the currency of the source account. For example, the currency for <BANKACCTFROM>.

The <CURRATE> should be the one in effect throughout the scope of the <CURRENCY> aggregate. It is not necessarily the current rate. Note that the <CURRATE> needs to take into account the choice of the FI for formatting of amounts (that is, where the decimal is) in both default and overriding currency, so that a client can do math. This can mean that the rate is adjusted by orders of magnitude (up or down) from what is commonly reported in newspapers.

Tag	*Description*
<CURRENCY> *or* <ORIGCURRENCY>	Currency aggregate
<CURRATE>	Ratio of <CURDEF> currency to <CURSYM> currency, in decimal form, ***rate***
<CURSYM>	ISO-4217 3-letter currency identifier, ***A-3***
</CURRENCY> *or*	

Tag	Description
</ORIGCURRENCY>	

In some cases, Open Financial Exchange defines transaction responses so that amounts have been converted to the home currency. However, Open Financial Exchange allows FIs to optionally report the original amount and the original (foreign) currency. In these cases, transactions include a specific tag for the original amount, and then a <ORIGCURRENCY> tag to report the details of the foreign currency.

Again, <CURRENCY> means that Open Financial Exchange *has not* converted amounts. Whereas, <ORIGCURRENCY> means that Open Financial Exchange *has* already converted amounts.

5.3 Country-Specific Tag Values

Some of the tags in Open Financial Exchange have values that are country-specific. For example, <USPRODUCTTYPE> is useful only within the United States. Open Financial Exchange will extend in each country as needed to provide tags that accept values useful to that country. Clients in other countries that do not know about these tags must simply skip them.

In some cases, a tag value represents a fundamental way of identifying something, yet there does not exist a world-wide standard for such identification. Examples include bank accounts and securities. In these cases, Open Financial Exchange must define a single, extensible approach for identification. For example, CUSIPs are used within the U.S., but not in other countries. However, CUSIPs are fundamental to relating investment securities, holdings, and transactions. Thus, a security ID consists of a two-part aggregate: one to identify the naming scheme, and one to provide a value. Open Financial Exchange will define valid naming schemes as necessary for each country.

6. Data Synchronization

6.1 Overview

Currently, some systems provide only limited support for error recovery and no support for backup files or multiple clients. The Open Financial Exchange data synchronization approach, described in this chapter, handles all of these situations.

Open Financial Exchange defines a powerful form of data synchronization between clients and servers.

Open Financial Exchange data synchronization addresses the following problems:

- Error recovery
- Use of multiple client applications
- Restoring from a backup file
- Multiple data files (for example, one copy at home, another at work).

This chapter first provides a brief background of error recovery problems and then presents the basic strategy used in Open Financial Exchange to perform data synchronization. Each Open Financial Exchange service includes specific details for data synchronization requests and responses.

Most of the information in this chapter concerns data synchronization, since it is a relatively new concept. The final section in this chapter discusses alternatives to full synchronization, and summarizes the options for each.

6.2 Background

When a client begins a connection with a server for which the connection does not successfully complete, there are two main problems:

- Unconfirmed requests

  If a client does not receive a response to work it initiates, it has no way of knowing whether the server processed the request. It also does not have any server-supplied information about the request, such as a server ID number.
- Unsolicited data

  Some banking protocols allow a server to send data to the client whenever the client makes a connection. This specification assumes that the first client that calls in after the unsolicited data is available for download receives the data. If the connection fails, this information would be forever lost to the client. Examples of unsolicited data include updates in the status of a bill payment and e-mail responses.

Unsolicited data presents problems beyond error recovery. Because the first client that connects to a server is the only one to receive unsolicited data, this situation precludes use of multiple clients without a data synchronization method. For example, if a user has a computer at work and one at home, and wants to perform online banking from both computers, a bank server could send unsolicited data to one but not the other.

An even greater problem occurs when a user resorts to a backup copy of the client data file. This backup file will be missing recent unsolicited data with no way to retrieve it from the server once the server sends it.

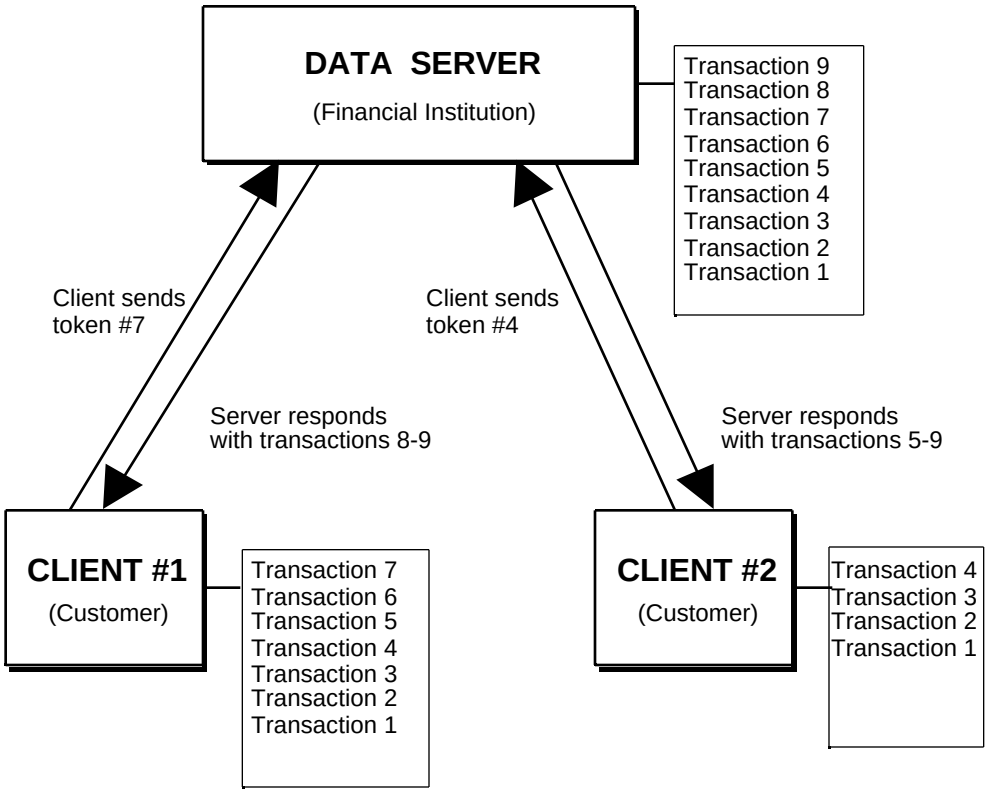
6.3 Data Synchronization Approach

A simple solution is to make sure that clients can always obtain information from the server for a reasonable length of time. Clients can request recent responses—whether due to client-initiated work or other status changes on the server—by supplying the previous endpoint in the response history. Servers always supply a new endpoint whenever they supply responses.

If a client connection fails—or a client receives a response, but crashes before updating its database—the client will not save the new endpoint. On the next synchronization request, the server sends the same information (plus any further status changes).

If a user switches to a backup file, again the client will use the older endpoint in the synchronization request.

If multiple clients are in use, each will send requests based on its own endpoint, so that both clients will obtain complete information from the server. This is one reason why Open Financial Exchange responses carry enough information from the request to enable them to be processed on their own. The diagram below shows the relationship between clients and servers.



Open Financial Exchange relieves the server from maintaining any special error-recovery state information. However, Open Financial Exchange requires the server to maintain a history of individual responses and a way to identify a position in the history. This ID could be a time stamp, or be based on its existing state information.

> ***Note:*** *Open Financial Exchange does not require servers to store responses based on individual connections. Also, not all requests are subject to synchronization. For example, Open Financial Exchange does not require servers to store statement-download responses separately for data synchronization.*

6.4 Data Synchronization Specifics

Open Financial Exchange does synchronization separately for each type of response. In addition, a synchronization request might include further identifying information, such as a specific account number. This specification defines the additional information for each synchronization request.

Each Open Financial Exchange service identifies the responses that are subject to data synchronization. For example, a bank statement-download is a read-only operation from the server. A client can request again if it fails; consequently, there is no data synchronization for this type of response.

The basis for synchronization is a *token* as defined by the <TOKEN> tag. The server can create a token in any way it wishes. The client simply holds the token for possible use in a future synchronization request.

The server can derive a token from one of the following:

- Time stamp
- Sequential number
- Unique non-sequential number
- Other convenient values for a server

> ***Note:*** *Open Financial Exchange reserves a <TOKEN> value of zero for the first time each type of response does a synchronization task.*

> ***Note:*** *Servers should return <TOKEN>=-1 (minus 1) in the event they must respond with an error. Since there is no XXXTRNRS wrapper inside the SYNCXXXRS wrapper, the <SYNCERROR> tag was added in version 2, to allow the server to return the value that would normally be in the <CODE> field of the <STATUS> aggregate.*

Clients will send a <TOKEN> of zero on their first synchronization request. Servers should send all available history, allowing a new client to know about work done by other clients. If a user's account has never been used with Open Financial Exchange, the server returns no history.

The server can use different types of tokens for different types of responses, if suitable for the server.

Tokens can contain up to 10 alphanumeric characters in V1 message sets, and 36 in V2 message sets; see Chapter 3, "Common Aggregates, Elements, and Data Types." Tokens need to be unique only with respect to the type of synchronization request and the additional information in that request. For example, a bill payment synchronization request takes an account number; therefore, a token needs to be unique only within payments for the account.

Servers will not have infinite history available, so synchronization responses can optionally include a <LOSTSYNC> element with a value of Y (yes) if the old token in the synchronization request was older than the earliest available history. This tag allows clients to alert users that some responses have been lost.

> ***Note:*** *Tokens are unrelated to <TRNUID>s, <SRVRTID>s, and <FITID>s, each of which serves a specific purpose and has its own scope and lifetime.*

A <SRVRTID> is not appropriate as a <TOKEN> for bill payment. A single payment has a single <SRVRTID>, but it can undergo several state changes over its life and thus have several entries in the token history.

There are three different ways a client and a server can conduct their requests and responses:

- Explicit synchronization – A client can request synchronization without sending any other Open Financial Exchange requests. The client sends a synchronization request, including the current token

for that request. The response includes responses more recent than the given token, along with a new token.

- Synchronization with new requests – A client can request synchronization as part any new requests it has. The client gives the old token. The response includes responses to the new requests plus any others that became available since the old token, along with a new token. An aggregate contains the requests so that the server can process the new requests and update the token as an atomic action.
- New requests without synchronization – A client can make new requests without providing the old token. In this case, it expects just responses to the new requests. A subsequent request for synchronization will cause the server to send the same response again, because the client did not update its token.

Each request and response that requires data synchronization will define a synchronization aggregate. The aggregate tells the server which kind of data it should synchronize. By convention, these aggregates always have SYNC as part of their tag names, for example, <PMTSYNCRQ>. You can use these aggregates on their own to perform explicit synchronization, or as wrappers around one or more new transactions. For example, you can use <PMTSYNCRQ> aggregates to request synchronization in combination with new work. You can use the <PMTTRNRQ> by itself if you do not require synchronization.

Some clients can choose to perform an explicit synchronization before sending any new requests (with or without synchronization). This practice allows clients to be up-to-date and possibly interact with the user before sending any new requests. Other clients can simply send new requests as part of the synchronization request.

If a client synchronizes in one file, then sends new work inside a synchronization request in a second file, there is a small chance that additional responses become available between the two connections. There is even a smaller chance that these would be conflicting requests, such as modifications to the same object. However, some clients and some requests might require absolute control, so that the user can be certain that they are changing known data. To support this, synchronization requests can optionally specify <REJECTIFMISSING> element. The tag tells a server that it should reject all enclosed requests if the supplied <TOKEN> is out of date *before considering the new requests.* That is, if any new responses became available, whether related to the incoming requests or not (but part of the scope of the synchronization request), the server should immediately reject the requests. It should still return the new responses. A client can then try again until it finds a stable window to submit the work. See section 6.5 for more information about conflict detection and resolution.

The password change request and response present a special problem. See section 2.5.2 for further information.

6.5 Conflict Detection and Resolution

Conflicts arise whenever two or more users or servers modify the same data. This can happen to any object that has a <SRVRTID> that supports change or delete requests. For example, one spouse and the other might independently modify the same recurring bill payment model. From a server perspective, there is usually no way to distinguish between the same user making two intended changes and two separate users making perhaps unintended changes. Therefore, Open Financial Exchange provides enough tools to allow clients to carefully detect and resolve conflicts. Open Financial Exchange requires only that a server process atomically all requests in a single <OFX> block.

A careful client always synchronizes before sending any new requests. If any responses come back that could affect a user's pending requests, the client can ask the user whether it should still send those pending requests. Because there is a small chance for additional server actions to occur between the initial synchronization request and sending the user's pending requests, extremely careful clients can use the <REJECTIFMISSING> element. Clients can iterate sending pending requests inside a synchronization request with <REJECTIFMISSING> and testing the responses to see if they conflict with pending requests. A client can continue to do this until a window of time exists wherein the client is the only agent trying to modify the server. In reality, this will almost always succeed on the first try.

6.6 Synchronization vs. Refresh

There are some situations, and some types of clients, for which it is preferable that the client ask the server to send everything it knows, rather than just a set of changes. For example, a client that has not connected often enough may have lost synchronization. Or, the user might be using a completely stateless client, such as a web browser.

> ***Note:*** *Open Financial Exchange does not require a client to refresh just because it has lost synchronization.*

Clients will mainly want to refresh lists of long-lived objects on the server; generally objects with a <SRVRTID>. For example, Open Financial Exchange Payments has both individual payments and models of recurring payments.

A brand new client, or a client that lost synchronization, might want to learn about in-progress payments by doing a synchronization refresh of the payment requests. It would almost certainly want to do a synchronization refresh of the recurring payment models, because these often live for months or years.

A client might not perform a synchronization refresh on e-mail responses.

A client can request a refresh by using the <REFRESH> tag with value of Y instead of the <TOKEN> tag. Server descriptions detail the exact behavior that servers should follow. However, the general rule is that servers should send responses that emulate a client creating or adding each of the objects governed by the particular synchronization request.

In these cases, servers set <TRNUID> to zero; the standard value for server-generated responses.

There is no need to recreate a stream of responses that emulate the entire history of the object, just an add response that reflects the current state. For example, if you create a model and then modify it three times, even if this history would have been available for a regular synchronization, servers should only send a single add that reflects the current state.

A client that wants only the current token, without refresh or synchronization, makes requests with <TOKENONLY> and a value of Y.

In all cases, servers should send the current ending <TOKEN> for the synchronization request in refresh responses. This allows a client to perform regular synchronization requests in the future.

The following table summarizes the options in a client synchronization request:

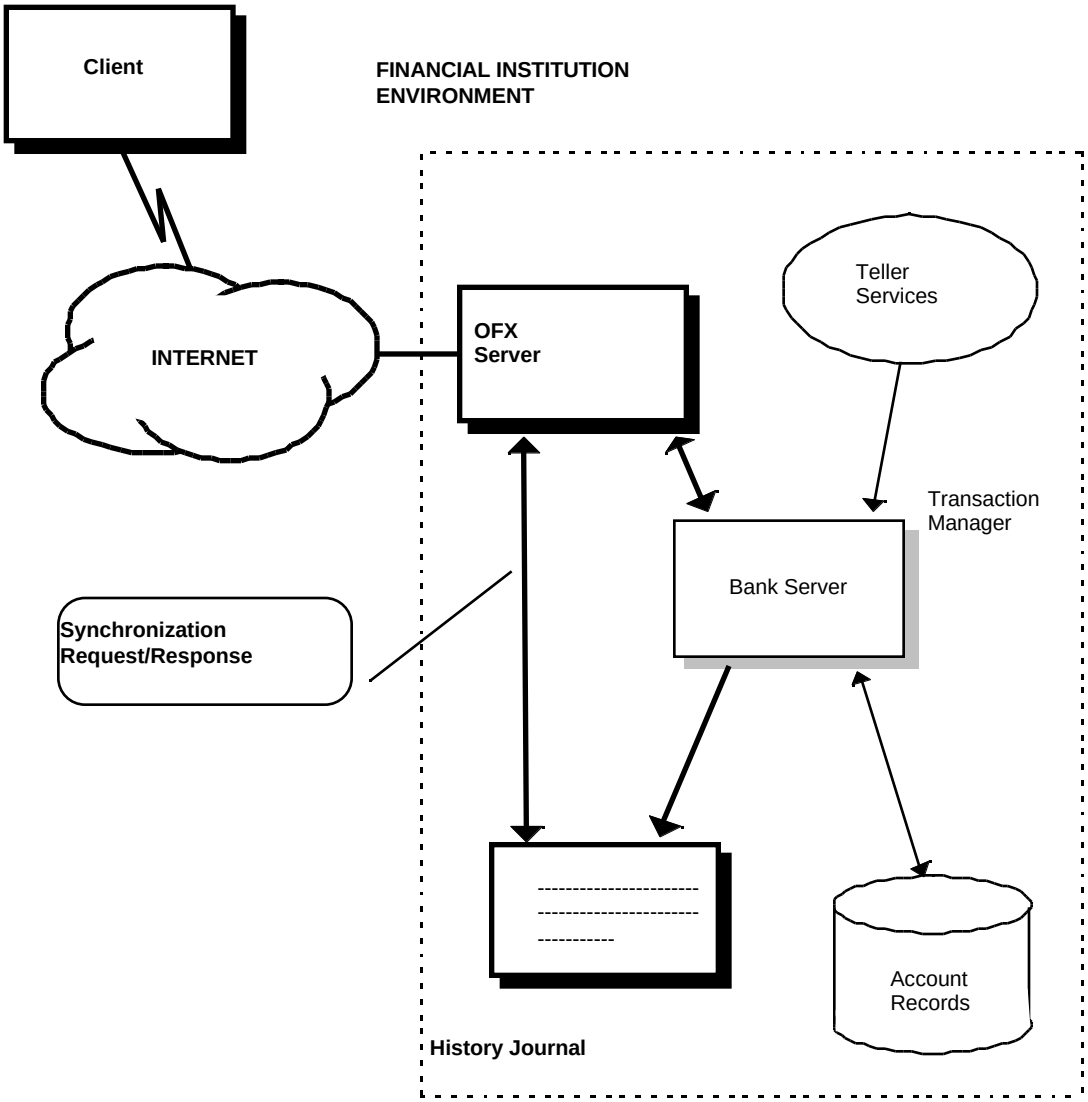
Tag	***Version***	***Description***
Client synchronization option; <TOKEN>, <TOKEN2>, <TOKENONLY>, or <REFRESH>		
<TOKEN>	V1 only	Previous value of <TOKEN> received for this type of synchronization request from server; 0 for first-time requests; ***token***
<TOKEN2>	V2 only	Previous value of <TOKEN2> received for this type of synchronization request from server; 0 for first-time requests; ***token2***
<TOKENONLY>		Request for just the current <TOKEN> without the history, ***Boolean***
<REFRESH>		Request for refresh of current state, ***Boolean***
<SYNCERROR>	V2 only	Optional error code, ***N-6***
		If token value is –1 to indicate that the sync request could not be processed, the server may return an error code with this tag. Codes are the same values that are used in the <CODE> field of the <STATUS> aggregate for transactions.
<REJECTIFMISSING>		If Y, do not process requests if client <TOKEN> is out of date, ***Boolean***

6.7 Typical Server Architecture for Synchronization

This section describes how an FI can approach supporting synchronization based on the assumption that modifications to an existing financial server will be kept to a minimum.

The simplest approach is to create a history database separate from the existing server. This history could consist of the actual Open Financial Exchange transaction responses (<TRNRS> aggregates) that are available to a synchronization request. The history database could index records by token, response type, and any other identifying information for that type, such as account number.

The diagram below shows a high-level model of the Open Financial Exchange architecture for a financial institution. Notice that the diagram shows the presence of a history journal.



The server adds responses to the history journal for any action that takes place on the existing server. This is true whether the Open Financial Exchange requests initiate the action or, in the case of recurring payments, it happens automatically on the server. Once added to the history journal, the server can forget them.

The areas of the Open Financial Exchange server that process synchronization requests need only search this history database for matching responses that are more recent than the incoming token.

For a refresh request, an Open Financial Exchange server would access the actual bank server to obtain the current state rather than recent history.

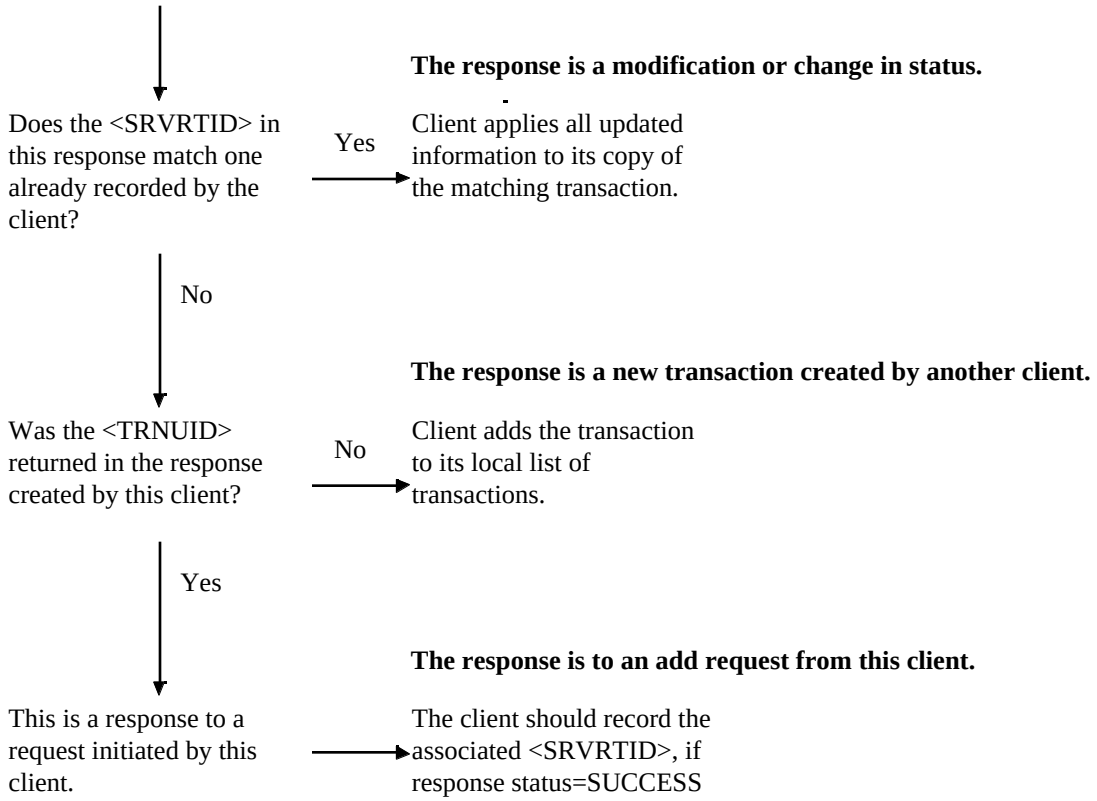
Periodically the bank server would purge the history server of older entries.

Only requests that are subject to synchronization need to have entries in the history database. Statement downloads do not involve synchronization; therefore, the FI server should not add these responses to the history database. Since statement downloads are usually the largest in space and the most frequent, eliminating these saves much of the space a response history might otherwise require.

More sophisticated implementations can save even more space. The history database could save responses in a coded binary form that is more compact than the full Open Financial Exchange response format. Some FIs might have much or all of the necessary data already in their servers; consequently, they would not require new data. An FI could regenerate synchronization responses rather than recall them from a database.

6.8 Typical Client Processing of Synchronization Results

The diagram below shows a general flowchart of what an Open Financial Exchange client would do with the results of a synchronization request. Most requests and responses subject to data synchronization contain both <TRNUID> and <SRVRTID>.



6.9 Simultaneous Connections

It is increasingly common that a server can get simultaneous or overlapping requests from the same user over two different computers. Open Financial Exchange requires a server to process each set of requests sent in a file as an atomic action. Servers can deal with the problems that arise with simultaneous use in two ways:

- Allow simultaneous connections, ensure each is processed atomically, and use the data synchronization mechanism to bring the two clients up to date. This is the preferred method.
- Lock out all but one user at a time, returning the error code 15501 for multiple users.

6.10 Synchronization Alternatives

Although it is **RECOMMENDED** that Open Financial Exchange servers implement full synchronization as described in this chapter, an alternate approach, "lite synchronization," could be easier for some servers to support. This approach focuses only on error recovery and does not provide any support for multiple clients, multiple data files, or use of backup files. The approach is to preserve the message sets while simplifying the implementation.

In addition, some clients might prefer to use response-file based error recovery with all servers, even if the client and some servers support full synchronization. This section first describes lite synchronization, and then explains the rules that clients and servers use to decide how to communicate.

> ***Note:*** *Lite synchronization servers do not support Refresh. In other words TOKEN=0 is not the same as Refresh synchronization. Therefore, clients should not send Refresh to a lite synchronization server.*

6.10.1 Lite Synchronization

Lite synchronization requires servers to accept all synchronization messages, but does not require them to keep any history or tokens. Responses need to be sent only once and then the server can forget them. Responses to client requests, whether or not they are made inside a synchronization request, are processed normally. Responses that represent server-initiated work, such as payment responses that arise from recurring payments, are sent only in response to synchronization requests. A server does not have to hold responses in case a second client makes a synchronization request.

Because full synchronization supports error recovery, an alternative is needed for lite synchronization. Servers using lite synchronization keep a copy of the entire response file they last sent. Clients requesting that servers prepare for error recovery generate a globally unique ID for each file they send. In the OFX headers, there are two tags associated with error recovery:

- OLDFILEUID – UID of the last request and response that was successfully received and processed by the client
- NEWFILEUID – UID of the current file

The format of these is the same as used with <TRNUID> as documented in Section 2.4.6.

Servers use the following rules:

- If OLDFILEUID is set to NONE, the client is not requesting file-based error recovery for this session. The server does not need to save the response file. In addition, since the client is not specifying a previous file that can now be committed, the server should not search for a response file to delete. Optionally, a server can choose to test for a client error by checking whether NEWFILEUID matches a previous request file. If NEWFILEUID matches a previous request file, but the client sent OLDFILEUID set to NONE, the client has committed an error. In this case, the server should report a general error.
- If OLDFILEUID is not set to NONE and NEWFILEUID matches a previous request file, the client is requesting error recovery. The server should send the matching saved response file.
- If OLDFILEUID is not set to NONE and OLDFILEUID matches a file saved on the server, then OLDFILEUID is a file that the client has successfully processed and the server can delete it. The client is also requesting that the response for the current file be saved under the NEWFILEUID for possible error recovery.

> ***Note:*** *If NEWFILEUID matches a previous request file then the request file identified by the NEWFILEUID must contain exactly the same set of transactions as the previous request file. Servers*

can reject the file if it contains new or modified transactions. Clients should disallow PINCH transactions during error recovery.

A server will never need to save more than one file per client data file, but because of possible multi-client or multi-data file usage, it might need to save several files for a given user. A server should save files for as long as possible, but not indefinitely. A server cannot recognize an error recovery attempt if it comes after it has purged the error recovery file. A server would process it as a new request. In this case, a server should recognize duplicate transaction UIDs for client-initiated work, such as payments, and then reject them individually. Server-generated responses would be lost to the client.

For a server accustomed to sending unsolicited responses, lite synchronization should closely match the current response-file based implementation. The only difference is that a server should hold the unsolicited responses until the client makes the first appropriate synchronization request; rather than automatically adding them to any response file. Once added, the server can mark them as delivered, relying on error recovery to insure actual delivery.

Note: *Open Financial Exchange requires a server to authenticate a client in Error Recovery.*

6.10.2 Relating Synchronization and Error Recovery

Client and server developers should first decide whether or not they will support full synchronization. If they can, then they can support response-file error recovery as well, or they can rely on synchronization to perform error recovery. If they adopt only lite synchronization, Open Financial Exchange requires response-file error recovery. A server describes each of these choices in its server profile records. The following combinations are valid:

- Full synchronization with response-file error recovery
- Full synchronization without separate response-file error recovery
- Lite synchronization with response-file error recovery

Clients request response-file error recovery by including the old and new session UIDs in the header. If they are absent, servers need not save the response file for error recovery. Clients request synchronization by using those synchronization requests defined throughout this specification.

6.11 Examples

Here is an example of full synchronization using bill payment as the service. Open Financial Exchange Payments provides two different synchronization requests and responses, each with their own token; one for payment requests and one for repeating payment model requests. Note that these simplified examples do not include the outer <OFX> layer, <SIGNON>, and so forth.

Client A requests synchronization:

```
<PMTSYNCRQ>
   <TOKEN>123
   <REJECTIFMISSING>N
   <BANKACCTFROM>
      <BANKID>121000248
      <ACCTID>123456789
      <ACCTTYPE>CHECKING
   </BANKACCTFROM>
</PMTSYNCRQ>
```

The server sends in response:

```
<PMTSYNCRS>
   <TOKEN>125
   <LOSTSYNC>N
   <BANKACCTFROM>
      <BANKID>121000248
      <ACCTID>123456789
      <ACCTTYPE>CHECKING
```

```
   </BANKACCTFROM>
   <PMTTRNRS>
      <TRNUID>123
      <STATUS>
         ... status details
      </STATUS>
      <PMTRS>
         ... details on a payment response
      </PMTRS>
   </PMTTRNRS>
   <PMTTRNRS>
      <TRNUID>546
      <STATUS>
         ... status details
      </STATUS>
      <PMTRS>
         ... details on another payment response
      </PMTRS>
   </PMTTRNRS>
</PMTSYNCRS>
```

Client A was missing two payment responses, which the server provides. At this point, client A is synchronized with the server. Client A now makes a new payment request, and includes a synchronization update as part of the request. This update avoids having to re-synchronize the expected response at a later time.

```
<PMTSYNCRQ>
   <TOKEN>125
   <REJECTIFMISSING>N
   <BANKACCTFROM>
      <BANKID>121000248
      <ACCTID>123456789
      <ACCTTYPE>CHECKING
   </BANKACCTFROM>
   <PMTTRNRQ>
      <TRNUID>12345
      <PMTRQ>
         ... details of a new payment request
      </PMTRQ>
   </PMTTRNRQ>
</PMTSYNCRQ>
```

The response to this new request:

```
<PMTSYNCRS>
   <TOKEN>126
   <LOSTSYNC>N
   <BANKACCTFROM>
      <BANKID>121000248
      <ACCTID>123456789
      <ACCTTYPE>CHECKING
   </BANKACCTFROM>
   <PMTTRNRS>
      ... details on a payment response to the new request
   </PMTTRNRS>
</PMTSYNCRS>
```

The client now knows that the server has processed the payments request it just made, and that nothing else has happened on the server since it last synchronized with the server.

Assume client B was synchronized with respect to payments for this account up through token 125. If it called in now and synchronized—with or without making additional requests—it would pick up the payment response associated with token 126. It records the same information that was in client A, which would give both clients a complete picture of payment status.

7. FI Profile

7.1 Overview

Open Financial Exchange clients use the profile to learn the capabilities of an Open Financial Exchange server. This information includes general properties such as account types supported, user password requirements, specific messages supported, and how the client should batch requests and where to send the requests. A client obtains a portion of the profile when a user first selects an FI. The client obtains the remaining information prior to sending any actual requests to that FI. The server uses a time stamp to indicate whether the server has updated the profile, and the client checks periodically to see if it should obtain a new profile.

In more detail, a profile response contains the following sections, which a client can request independently:

- Message Sets – list of services and any general attributes of those services. Message sets are collections of messages that are related functionally. They are generally subsets of what users see as a service.
- Signon realms – FIs can require different signons (user ID and/or password) for different message sets. Because there can only be one signon per <OFX> block, a client needs to know which signon the server requires and then provide the right signon for the right batch of messages.

The profile message is itself a message set. In files, Open Financial Exchange uses the <PROFMSGSV1> and <PROFMSGSV2> aggregates to identify this profile message set.

The following sections describe the general use of profile information.

7.1.1 Message Sets

A message set may be thought of as representing an available financial service. A message set itself is a collection of related messages. For example, Chapter 11, "Banking," defines several message sets: statement download, credit card statement download, intrabank transfers, and so forth. A server may route all of the messages in a message set to a single URL and merge their versions together.

Clients and servers generally use message sets as the granularity to decide what functionality they will support. A "banking" server can choose to support the statement download and intrabank transfer message sets, but not the wire transfer message set. Attributes are available in many cases to further define how Open Financial Exchange supports a message set.

Each portion of the Open Financial Exchange specification that defines messages also defines the message set to which that the messages belongs. This includes what additional attributes are available for those messages and whether Open Financial Exchange requires the message set or it is optional.

7.1.2 Version Control

Message sets are the basis of version control. Over time there will be new versions of the message sets, and at any given time servers will likely want to support more than one version of a message set. Clients should also be capable of supporting as many versions as possible. Through the profile, clients discover which versions are supported for each message set. Clients and servers exchange messages at the highest common level for each message set.

For the Open Financial Exchange-SGML data format, there is a single DTD for all message sets. The version number of the DTD advances when any *syntactic* change is made to any of the message sets. (It is possible to make a *semantic* change that would not even require a change in syntax. A change in rules, for example, that would change the version of the message set without changing the DTD.) A single DTD cannot have two different definitions for the same aggregate. There are limitations to how a server that uses true DTD-based parsing can handle multiple versions of a message at the same time.

Version 2 of the profile message set allows 1 or more <MSGSETCORE> aggregates to be returned to the client. This capability allows for multiple URLs to be utilized for different versions of each <MSGSETCORE>.

Version 2 of the profile message set also allows 1 or more <URL> elements in <MSGSETCORE>, where version 1 of the profile message set allowed only 1 <URL> element. This was an oversight in version 1 of <MSGSETCORE>. The need for multiple URLs to be listed existed for signon even when there was only one version of each message set.

The purpose of more than 1 URL for a single version of a message set is to allow the same version of signon to be sent to all the URLs that require that version of signon. For example, if banking version 1 is at one URL (A) and billpay version 1 is at another URL (B), both may need version 1 of signon to be used. In that case, <MSGSETCORE> inside <BANKMSGSETV1> would refer only to <URL> A and <MSGSETCORE> inside <BILLPAYMSGSETV1> would refer only to <URL> B, but both URLs (A and B) would be listed inside <MSGSETCORE> inside <SIGNONMSGSETV1>.

7.1.3 Batching and Routing

To allow FIs to set up different servers for different message sets, different versions, or to directly route some messages to third party processors, message sets define the URL to which a server sends messages in that message set. Each version of a message set can have a different URL. In the common case where many or all message sets are sent to a single URL, clients will consolidate messages across compatible message sets. Clients can consolidate when:

- Message sets have the same URL

- Message sets have a common security level
- Message sets have the same signon realm

Note: *The signon message set can be used with other message sets even if it contains incompatible settings for the URL, security level, or signon realm. The message set information for signon messages is used only if the signon message is sent by itself.*

The same message set may be supported by multiple servers. In this case, each server that supports a particular message set must have a unique URL.

7.1.4 Client Signon for Profile Requests

Clients must include a signon request <SONRQ> with every message, including profile requests. The first time that a client requests the FI profile, the signon request will be present, but the user ID and password will not be valid and will be ignored by the server.

Note: *Since elements cannot be set to a blank value, <USERID> and/or <USERPASS> may be set to lower case "anonymous" followed by 23 zeroes.*

Once the user has enrolled and received his or her user ID and password, the client will request the profile again, even if the profile is not yet out-of-date. At this point, the server can respond with a profile response that indicates that the profile is up-to-date. Alternatively, if the FI wants to return a customer-specific profile, the FI can choose to return a new FI profile in the response.

For more information about signon requests, refer to section 2.5.

7.1.5 Profile Request <PROFRQ>

A profile request indicates which profile components a client desires. It also indicates what the client's routing capability is. Profiles returned by the FI must be compatible with the requested routing style, or the server returns an error.

Profile requests are not subject to synchronization.

Profile requests must appear within a <PROFTRNRQ> transaction wrapper.

Tag	*Description*
<**PROFRQ**>	Profile-request aggregate
<CLIENTROUTING>	Identifies client routing capabilities, see table below
<DTPROFUP>	Date and time client last received a profile update, ***datetime***
</PROFRQ>	

Tag	*Description*
NONE	Client cannot perform any routing. All URLs must be the same. All message sets share a single signon realm.
SERVICE	Client can perform limited routing. See details below.
MSGSET	Client can route at the message-set level. Each message set can have a different URL and/or signon realm.

The SERVICE option supports clients that can route bill payment messages to a separate URL from the rest of the messages. Because the exact mapping of message sets to the general concept of bill payment can vary by client and by locale, this specification does not provide precise rules for the SERVICE option. Each client will define its requirements.

7.2 Profile Response <PROFRS>

To determine whether the client has the latest version of the FIs profile, the server checks the date and time passed by the client in <DTPROFUP>.

If the client has the latest version of the FIs profile, the server returns status code 1 in the <STATUS> aggregate of the profile-transaction aggregate <PROFTRNRS>. The server does not return a profile-response aggregate <PROFRS>.

If the client does not have the latest version of the FI profile, the server responds with the profile-response aggregate <PROFRS> in the profile-transaction aggregate <PROFTRNRS>. The response includes message set descriptions, signon information, and general contact information.

Tag	*Version*	*Description*
<PROFRS>		Profile-response aggregate
<MSGSETLIST>		Beginning list of message set information
<XXXMSGSET>		One or more message set aggregates
</XXXMSGSET>		
</MSGSETLIST>		
<SIGNONINFOLIST>		Beginning of signon information
<SIGNONINFO>		Zero or more signon information aggregates
</SIGNONINFO>		
</SIGNONINFOLIST>		
<DTPROFUP>		Time this was updated on server, ***datetime***
<FINAME>		Name of institution, ***A-32***
<ADDR1>		FI address, line 1, ***A-32***
<ADDR2>		FI address, line 2, ***A-32***
<ADDR3>		FI address, line 3, ***A-32***
<CITY>		FI address city, ***A-32***
<STATE>		FI address state, ***A-5***
<POSTALCODE>		FI address postal code, ***A-11***
<COUNTRY>		FI address country; 3-letter country code from ISO/DIS-3166, ***A-3***
<CSPHONE>		Customer service telephone number, ***A-32***
<TSPHONE>		Technical support telephone number, ***A-32***
<FAXPHONE>		Fax number, ***A-32***
<URL>	V1 only	URL for general information about FI (not for sending data), ***URL***
<URL2>	V2 only	URL for general information about FI (not for sending data), ***URL2***
<URLGETREDIRECT>	V2 only	Whether or not the returned URL is a redirection URL. ***Boolean***
<EMAIL>		E-mail address for FI, ***A-80***

Tag	Version	Description
</PROFRS>		

7.2.1 Message Set

An aggregate describes each message set supported by an FI. Message sets in turn contain an aggregate for each version of the message set that is supported. For a message set named *XXX*, the convention is to name the outer aggregate <*XXX*MSGSET> and the tag for each version <*XXX*MSGSETVn>. The reason for message set-specific aggregates is that the set of attributes depends on the message set. These can change from version to version, so there are version-specific aggregates as well.

The general form of the response is:

Tag	Description
<XXXMSGSET>	Service aggregate
<XXXMSGSETVn>	Version-of-message-set aggregate, 1 or more
</XXXMSGSETVn>	
</XXXMSGSET>	

The <***XXX*MSGSETVn**> aggregate has the following form:

Tag	Description
<*XXX*MSGSETVn>	Message-set-version aggregate
<MSGSETCORE>	1 or more common message set information aggregates
</MSGSETCORE>	End tag for each message set information aggregate
Message-set specific	Zero or more attributes specific to this version of this message set, as defined by each message set
</*XXX*MSGSETVn>	

The common message set information <MSGSETCORE> is as follows:

Tag	Version	Description
<MSGSETCORE>		Common-message-set-information aggregate
<VER>		Version number of the message set, (for example, <VER>1 for version 1 of the message set), ***N-5***
<URL>	V1 only	URL where messages in this set are to be sent, ***URL***
<URL2>	V2 only	URL where messages in this set are to be sent, ***URL2***
<OFXSEC>		Security level required for this message set; see Chapter 4, "Open Financial Exchange Security." ***NONE or TYPE 1.***
<TRANSPSEC>		Y if transport-level security must be used, N if not used; see Chapter 4, "Open Financial Exchange Security." ***Boolean***
<SIGNONREALM>		Signon realm to use with this message set, ***A-32***
<LANGUAGE>		Language supported, ***language.*** If more than one language is supported, multiple <LANGUAGE> tags can be sent.
<PREFIX**>**	V2 only	Prefix of country-specific tags supported and field interpretations: 3-letter country code from ISO/DIS-3166. If multiple <PREFIX> tags are present, then multiple countries are supported. If this field is missing, then it is assumed that only the USA

Tag	Version	Description
		version of OFX is supported. For example, if the values “FRA” and “ITA” are present, then the server supports the French and Italian “versions” of OFX. If a country is supported for any of the main message sets, then the server should accept a signon request for that country. Otherwise, the server should reject such a signon request. The following countries are currently supported in OFX: PREFIX Country Name BEL Belgium CAN Canada CHE Switzerland DEU Germany ESP Spain FRA France GBR Great Britain ITA Italy NLD Netherlands USA United States of America
<SYNCMODE>		FULL for full synchronization capability LITE for lite synchronization capability See Chapter 6, “Data Synchronization,” for more information.
<RESPFILEER>		Y if server supports response-file based error recovery, ***Boolean*** See Chapter 6, “Data Synchronization,” for more information.
<SPNAME>		Service provider name, ***A-32*** ***Some financial institutions out-source their OFX servers to a service provider. In such cases, the SPNAME element should be included in the MSGSETCORE.***
</MSGSETCORE>		

Note: *For all message sets currently defined in Open Financial Exchange, the value of <TRANSPSEC> must be Y.*

Note: *Within a <MSGSETCORE> aggregate, the <VER> element defines the version number of that message set. It does not refer to the version number of the Open Financial Exchange specification or the DTD files. For more information about message sets and version numbers, refer to section 2.4.5.*

Note: *Within a message set, there can be more than one <MSGSETCORE> aggregate with the same value for <VER>, or the same value for <URL>, but not the same value for both. The pair must be unique for each instance of <MSGSETCORE> within a message set. Multiple <MSGSETCORE>s with the same value for <VER> are used in instances such as signon or registration, which may have the same version sent to multiple URLs for different services.*

7.2.2 Signon Realms

A signon realm identifies a set of messages that can be accessed using the same password. Realms are used to disassociate signons from specific services, allowing FIs to require different signons for different message sets. In practice, FIs will want to use the absolute minimum number of realms possible to reduce the user’s workload.

Tag	Version	Description
<SIGNONINFO>		Signon-information aggregate
<SIGNONREALM>		Identifies this realm, ***A-32***
<MIN>		Minimum number of password characters, ***N-2***

Tag	Version	Description
<MAX>		Maximum number of password characters, ***N-2***
<CHARTYPE>		Type of characters allowed in password; one of ALPHAONLY, NUMERICONLY, ALPHAORNUMERIC, or ALPHAANDNUMERIC.
<CASESEN>		Y if password is case-sensitive, ***Boolean***
<SPECIAL>		Y if special characters are allowed, ***Boolean***
<SPACES>		Y if spaces are allowed, ***Boolean***
<PINCH>		Y if server supports USERPASSPIN-change requests, ***Boolean***
<CHGPINFIRST>		Y if server requires clients to change USERPASSPIN as part of first signon, ***Boolean***
<PWTYPE**>**	V2 only	The type of password(s) supplied by the client in the signon request to authenticate the user (see section 2.5.1.1). The following password types are currently supported in OFX: PWTYPE — Description FIXED — The client supplies (in USERPASS) a normal password that remains fixed until explicitly changed by the user or the FI. ONETIME — In addition to USERPASS, the client supplies (in ONETIMEPASS) an additional password that was taken from a list of one-time passwords sent from the FI to the user. HWTOKEN — In addition to USERPASS, the client supplies (in ONETIMEPASS) an additional password that was generated by a hardware token such as a crypto-calculator, typically on a time-varying basis. If the PWTYPE tag is not present, then the FIXED password type is assumed as the default.
</SIGNONINFO>		

7.2.3 Status Codes

Value	Meaning
0	Success (INFO)
1	Client is up-to-date (INFO)
2000	General error (ERROR)

7.3 Profile Message Set Profile Information

The profile message set functions the same way as all other message sets; therefore, it contains a profile description for that message set. Because <PROFMSGSET> is always part of a message set response, it is described here. Servers must include the <PROFMSGSET> as part of the profile response <MSGSETLIST>. There are no attributes, but the aggregate must be present to indicate support for the message set.

Tag	Description
<PROFMSGSET>	Message-set-profile-information aggregate
<PROFMSGSETV1>	Opening tag for V1 of the message set profile information
<MSGSETCORE>	Common message set information
</MSGSETCORE>	
</PROFMSGSETV1>	

Tag	*Description*
<PROFMSGSETV2>	Opening tag for V2 of the message set profile information
<MSGSETCORE>	Common message set information
</MSGSETCORE>	
</PROFMSGSETV2>	
</PROFMSGSET>	

8. Activation & Account Information

8.1 Overview

The Signup message set defines three messages to help users get setup with their FI:

- Enrollment – informs FI that a user wants to use Open Financial Exchange and requests that a password be returned
- Accounts – asks the FI to return a list of accounts and the services supported for each account
- Activation – allows a client to tell the FI which services a user wants on each account

There is also a message to request name and address changes.

Clients use the account information request on a regular basis to look for changes in a user's account information. A time stamp is part of the request so that a server has to report only new changes. Account activation requests are subject to data synchronization, and will allow multiple clients to learn how the other clients have been enabled.

In Open Financial Exchange files, the <SIGNUPMSGSV1> and <SIGNUPMSGSV2> aggregates identify the Signup message.

8.2 Approaches to User Sign-Up with Open Financial Exchange

The message sets in this chapter are designed to allow both FIs and clients to support a variety of sign-up procedures. There are four basic steps a user needs to go through to complete the sign-up:

1. **Select the FI**. Open Financial Exchange does not define this step or provide message sets to support it. Client developers and FIs can let a user browse or search this information on a web site, or might define additional message sets to do this within the client. At the conclusion of this step, the client will have some minimal profile information about the FI, including the set of services supported and the URL to use for the next step.
2. **Enrollment and password acquisition.** In this step, the user identifies and authenticates itself to the FI *without a password*. In return, the user obtains a password (possibly temporary) to use with Open Financial Exchange. FIs can perform this entire step over the telephone, through a combination of telephone requests and a mailed response, or at the FI web site. FIs can also use the Open Financial Exchange enrollment message to do this by means of the client. The response can contain a temporary password or users can wait for a mailed welcome letter containing the password.
3. **Account Information.** In this step, the user obtains a list of accounts available for use with Open Financial Exchange, and which specific services are available for each account. Even if users have enrolled over the telephone, clients will still use this message set to help users properly set up the accounts within the client. Clients periodically check back with the FI for updates.
4. **Service Activation.** The last step is to activate specific services on specific accounts. The activation messages support this step. Synchronization is applied to these messages to ensure that other clients are aware of activated services.

The combination of media-interface through which an FI accomplishes these steps can vary. FIs might wish to do steps two through four over the telephone. Clients will still use Open Financial Exchange messages in steps 3 and 4 to automatically set up the client based on the choices made by the user over the phone. Other FIs might wish to have the entire user experience occur within the client. Either way, the Open Financial Exchange sign-up messages support the process.

8.3 Users and Accounts

To support the widest possible set of FIs, Open Financial Exchange assumes that individual users and accounts are in a many-to-many relationship. Consider a household with three accounts:

- Checking 1 – held individually by one spouse
- Checking 2 – held jointly by both
- Checking 3 – held individually by the other spouse

Checking 2 should be available to either spouse, and the spouse holding Checking 1 should be able to see both Checking 1 and 2.

Open Financial Exchange expects FIs to give each user their own user ID and password. Each user will go through the enrollment step separately. A given account need only be activated once for a service; not once for each user. Clients will use the account information and activation messages to combine information about jointly held accounts.

If an FI prefers to have a single user ID and password per household or per master account, it will have to make this clear to users through the enrollment process. It is up to the FI to assign a single user ID and password that can access all three of the checking accounts described above.

8.4 Enrollment and Password Acquisition

The main purpose of the enrollment message is to communicate a user's intent to access the FI by way of Open Financial Exchange and to acquire a password for future use with Open Financial Exchange. Some FIs might return a user ID and an initial password in the enrollment response, while others will send them by way of regular mail.

> ***Note:*** *The client may not know the user ID and password when the it sends the enrollment request, in such a case the <USERID> and/or <USERPASS> may be set to lower case "anonymous" followed by 23 zeroes.*

Enrollment requests are not subject to synchronization. If the client does not receive a response, it will simply re-request the enrollment. If a user successfully enrolls from another client before the first client obtains a response, the server should respond to subsequent requests from the first client with status code:

13501 - user already enrolled.

8.4.1 User IDs

The Open Financial Exchange <SONRQ> requires a user ID to uniquely identify a user to an FI. The server must accept the user ID with or without punctuation.

Many FIs in the United States use social security numbers (SSNs) as the ID. Others create IDs that are unrelated to the users' SSNs. Some FIs have existing user IDs that they use for other online activities that they want to use for Open Financial Exchange as well. FIs might also create new IDs specifically for Open Financial Exchange. Finally, some FIs might assign IDs while others might allow users to create them. Because users do not usually know either their Open Financial Exchange sign-on user ID or their password at time of enrollment, the enrollment response is designed to return both. The enrollment request allows users to optionally provide a user ID, which an FI can interpret as their existing online ID or a suggestion for what their new user ID should be. Ideally, the enrollment process should explain ID syntax to users.

8.4.2 Enrollment Request <ENROLLRQ>

The enrollment request captures enough information to identify and authenticate a user as being legitimate and that it has a relationship with the FI.

FIs might require that an account number be entered as part of the identification process. However, this is discouraged since the account information request is designed to automatically obtain all account information, avoiding the effort and potential mistakes of a user-supplied account number.

It is **RECOMMENDED** that FIs provide detailed specifications for user IDs and passwords along with information about the services available when a user is choosing an FI.

The enrollment request must appear within an <ENROLLTRNRQ> transaction wrapper.

Tag	*Description*
<ENROLLRQ>	Enrollment-request aggregate
<FIRSTNAME>	First name of user, ***A-32***
<MIDDLENAME>	Middle name of user, ***A-32***
<LASTNAME>	Last name of user, ***A-32***
<ADDR1>	Address line 1, ***A-32***
<ADDR2>	Address line 2, ***A-32***
<ADDR3>	Address line 3, ***A-32***
<CITY>	City, ***A-32***
<STATE>	State or province, ***A-5***
<POSTALCODE>	Postal code, ***A-11***
<COUNTRY>	3-letter country code from ISO/DIS-3166, ***A-3***
<DAYPHONE>	Daytime telephone number, ***A-32***
<EVEPHONE>	Evening telephone number, ***A-32***
<EMAIL>	Electronic e-mail address, ***A-80***
<USERID>	Actual user ID if already known, or preferred user ID if user can pick, ***A-32***
<TAXID>	ID used for tax purposes (such as SSN), may be same as user ID, ***A-32***
<SECURITYNAME>	Mother's maiden name or equivalent, ***A-32***

Tag	Description
<DATEBIRTH>	Date of birth, ***date***
<***XXX***ACCTFROM>	An account description aggregate for an existing account at the FI, for identification purposes only. For example, <BANKACCTFROM> or <INVACCTFROM>.
</***XXX***ACCTFROM>	
</ENROLLRQ>	

This enrollment request is intended for use only by individuals. Business enrollment will be defined in a later release.

8.4.3 Enrollment Response <ENROLLRS>

The main purpose of the enrollment response is to acknowledge the request. In those cases where FIs permit delivery of an ID and a temporary password, the response also provides for this. Otherwise the server will send the real response to the user by way of regular mail, electronic mail, or over the telephone. If enrollment is successful, but the server does not return the ID and password in the response, a server is REQUIRED to use status code 10 and provide some information to the user by means of the <MESSAGE> element in the <STATUS> aggregate about what to expect next.

The enrollment response must appear within an <ENROLLTRNRS> transaction wrapper.

Tag	Description
<ENROLLRS>	Enrollment-response aggregate
<TEMPPASS>	Temporary password, ***A-32***
<USERID>	User ID, ***A-32***
<DTEXPIRE>	Time the temporary password expires (if <TEMPPASS> included), ***datetime***
</ENROLLRS>	

8.4.4 Enrollment Status Codes

Code	Meaning
0	Success (INFO)
2000	General error (ERROR)
13000	User ID & password will be sent out-of-band (INFO)
13500	Unable to enroll user (ERROR)
13501	User already enrolled (ERROR)

8.4.5 Examples

An enrollment request:

```
<ENROLLTRNRQ>
   <TRNUID>12345
   <ENROLLRQ>
      <FIRSTNAME>Joe
      <MIDDLENAME>Lee
      <LASTNAME>Smith
      <ADDR1>21 Main St.
      <CITY>Anytown
      <STATE>TX
```

```
      <POSTALCODE>87321
      <COUNTRY>USA
      <DAYPHONE>123-456-7890
      <EVEPHONE>987-654-3210
      <EMAIL>jsmith@isp.com
      <USERID>jls
      <TAXID>123-456-1234
      <SECURITYNAME>jbmam
      <DATEBIRTH>19530202
   </ENROLLRQ>
</ENROLLTRNRQ>
```

And the reply might be:

```
<ENROLLTRNRS>
   <TRNUID>12345
   <STATUS>
      <CODE>0
      <SEVERITY>INFO
   </STATUS>
   <ENROLLRS>
      <TEMPPASS>changeme
      <USERID>jls
      <DTEXPIRE>19970105
   </ENROLLRS>
</ENROLLTRNRS>
```

8.5 Account Information

Account information requests ask a server to identify and describe all of the accounts accessible by the signed-on user. The definition of *all* is up to the FI. At a minimum, it is **RECOMMENDED** that a server include information about all accounts that it can activate for one or more Open Financial Exchange services. To give the user a complete picture of his relationship with an FI, FIs can give information on other accounts, even if those accounts are available only for limited Open Financial Exchange services.

Some service providers do not have prior knowledge of user account information. The profile allows these servers to report this, and clients then know to ask users for account information rather than reading it from the server.

Clients can perform several tasks for users with this account information. First, the information helps a client set up a user for online services by giving it a precise list of its account information and available services for each. Clients can set up their own internal state as well as prepare service activation requests with no further typing by users. This can eliminate data entry mistakes in account numbers, routing transit numbers, and so forth.

Second, FIs can provide limited information on accounts that would not ordinarily be suitable to Open Financial Exchange services. For example, a balance-only statement download would be useful for certificates of deposits even though a customer or an FI might not want or allow CDs to be used for full statement download.

For each account, there is one <ACCTINFO> aggregate returned. The aggregate includes one service-specific account information aggregate for each service available to that account. That, in turn, provides the service-specific account identification. Common to each service-specific account information aggregate is the <SVCSTATUS> tag, which indicates the status of this service on this account.

A server should return joint accounts (accounts for which more than one user ID can be used to access the account) for either user.

Requests and responses include a <DTACCTUP> element. Responses contain the last time a server updated the information. Clients *are* **REQUIRED** *to* send this in a subsequent request, and servers are **REQUIRED** to compare this to the current modification time and only send information if it is more recent. The server sends the entire account information response if the client's time is older; there is no attempt to incrementally update specific account information.

8.5.1 Request <ACCTINFORQ>

The <ACCTINFORQ> request must appear within an <ACCTINFOTRNRQ> transaction wrapper.

Tag	*Description*
<ACCTINFORQ>	Account-information-request aggregate
<DTACCTUP>	Last <DTACCTUP> received in a response, ***datetime***
</ACCTINFORQ>	

8.5.2 Response <ACCTINFORS>

The <ACCTINFORS> response must appear within an <ACCTINFOTRNRS> transaction wrapper.

Tag	*Description*
<ACCTINFORS>	Account-information-response aggregate
<DTACCTUP>	Date and time of last update to this information on the server, ***datetime***
<ACCTINFO>	Zero or more account information aggregates

Tag	Description
</ACCTINFO>	
</ACCTINFORS>	End of account information response

8.5.3 Account Information Aggregate <ACCTINFO>

Tag	Description
<ACCTINFO>	Account-information-record aggregate
<DESC>	Description of the account, ***A-80***
<PHONE>	Telephone number for the account, ***A-32***
<XXXACCTINFO>	Service-specific account information, defined in each service chapter. For a given service ***XXX***, there can be at most one <***XXX***ACCTINFO> returned. For example, you cannot return two <BANKACCTINFO> aggregates.
<XXXACCTFROM>	Service-specific account identification
</XXXACCTFROM>	
<SVCSTATUS>	AVAIL = Available, but not yet requested PEND = Requested, but not yet available ACTIVE = In use
</XXXACCTINFO>	
</ACCTINFO>	

Note: *A server uses the <DESC> field to convey the FI's preferred name for the account, such as "PowerChecking." It should not include the account number.*

8.5.4 Status Codes

Code	Meaning
0	Success (INFO)
1	Client is up-to-date (INFO)
2000	General error (ERROR)

8.5.5 Examples

An account information request:

```
<ACCTINFOTRNRQ>
   <TRNUID>12345
   <ACCTINFORQ>
      <DTACCTUP>19960101

   </ACCTINFORQ>
</ACCTINFOTRNRQ>
```

And a response for a user with access to one account, supporting banking:

```
<ACCTINFOTRNRS>
   <TRNUID>12345
   <STATUS>
      <CODE>0
```

```
      <SEVERITY>INFO
   </STATUS>
   <ACCTINFORS>
      <DTACCTUP>19960102
      <ACCTINFO>
      <DESC>Power Checking
      <PHONE>8002223333

         <BANKACCTINFO>
            <BANKACCTFROM>
               <BANKID>1234567789
               <ACCTID>12345
               <ACCTTYPE>CHECKING
            </BANKACCTFROM>
         <SUPTXDL>Y
         <XFERSRC>Y
         <XFERDEST>Y
         <SVCSTATUS>ACTIVE
         </BANKACCTINFO>
      </ACCTINFO>
   </ACCTINFORS>
</ACCTINFOTRNRS>
```

8.6 Service Activation

Clients inform FIs that they wish to start, modify, or terminate a service for an account by sending service activation requests. These are subject to data synchronization, and servers should send responses to inform clients of any changes, even if the changes originated on the server.

Clients use these records during the initial user sign-up process. Once a client learns about the available accounts and services (by using the account information request above, or by having a user directly enter the required information), it sends a series of service ADD requests.

If a user changes any of the identifying information about an account, the client sends a service activation request containing both the old and the new account information. Servers should interpret this as a change in the account, not a request to transfer the service between two existing accounts, and all account-based information such as synchronization tokens should continue. If a user or FI is reporting that service should be moved between two existing accounts, service must be terminated for the old account and started for the new account. The new account will have reset token histories, as with any new service.

Each service to be added, changed, or removed is contained in its own request because the same real-world account might require different <*XXX*ACCTFROM> aggregates depending on the type of service.

8.6.1 Activation Request and Response

8.6.1.1 Request <ACCTRQ>

The <ACCTRQ> request must appear within an <ACCTTRNRQ> transaction wrapper.

Tag	*Version*	*Description*
<ACCTRQ>		Account-service-request aggregate
Action identification. . Specify either <SVCADD>, <SVCCHG>, or <SVCDEL>		Action aggregate, either <SVCADD>, <SVCCHG>, or <SVCDEL>
<SCVCADD>		Service-addition aggregate
</SCVCADD>		
-or-		
<SVCCHG>		Service-change aggregate

Tag	Version	Description
</SVCCHG>		
-or-		
<SVCDEL>		Service-deletion aggregate
</SVCDEL>		
<SVC>	V1 only	Service to be added/changed/deleted BANKSVC = Banking service BPSVC = Payments service INVSVC = Investments
<SVC2>	V2 only	Service to be added/changed/deleted BANKSVC = Banking service BPSVC = Payments service INVSVC = Investments PRESSVC = Bill Presentment service
</ACCTRQ>		

8.6.1.2 Response <ACCTRS>

The <ACCTRS> response must appear within an <ACCTTRNRS> transaction wrapper.

Tag	Version	Description
<ACCTRS>		Account-service-response aggregate
Action identification. Specify either <SVCADD>, <SVCCHG>, or <SVCDEL>		
<SCVCADD>		Service-addition aggregate
</SCVCADD>		
-or-		
<SVCCHG>		Service-change aggregate
</SVCCHG>		
-or-		
<SVCDEL>		Service-deletion aggregate
</SVCDEL>		
<SVC>	V1 only	Service to be added/changed: BANKSVC = Banking service BPSVC = Payments service INVSVC = Investments
<SVC2>	V2 only	Service to be added/changed: BANKSVC = Banking service BPSVC = Payments service INVSVC = Investments PRESSVC = Bill Presentment service
<SVCSTATUS>		AVAIL = Available, but not yet requested PEND = Requested, but not yet available ACTIVE = In use
</ACCTRS>		

8.6.1.3 Service Add Aggregate <SVCADD>

Tag	*Version*	*Description*
<SVCADD>		Service-addition aggregate
<*XXX*ACCTTO>		Service-specific-account-identification aggregate (for example, <BANKACCTTO> or <INVACCTTO>)
</*XXX*ACCTTO>		
<PREAUTHTOKEN>	V2 only	A token provided out-of-band by an FI or biller to speed up activation of the account, ***A-32***
</SVCADD>		

8.6.1.4 Service Change Aggregate <SVCCHG>

Tag	*Description*
<SVCCHG>	Service-change aggregate
<*XXX*ACCTFROM>	Service-specific-account-identification aggregate (for example, <BANKACCTFROM> or <INVACCTFROM>)
</*XXX*ACCTFROM>	
<*XXX*ACCTTO>	Service-specific-account-identification aggregate (for example, <BANKACCTTO> or <INVACCTTO>)
</*XXX*ACCTTO>	
</SVCCHG>	

8.6.1.5 Service Delete Aggregate <SVCDEL>

Tag	*Description*
<SVCDEL>	Service-deletion aggregate
<*XXX*ACCTFROM>	Service-specific-account-identification aggregate (for example, <BANKACCTFROM> or <INVACCTFROM>)
</*XXX*ACCTFROM>	
</SVCDEL>	

8.6.1.6 Status Codes

Code	*Meaning*
0	Success (INFO)
2000	General error (ERROR)
2002	General account error (ERROR)
2006	Source account not found (ERROR)
2007	Source account closed (ERROR)
2008	Source account not authorized (ERROR)
2009	Destination account not found (ERROR)
2010	Destination account closed (ERROR)

Code	Meaning
2011	Destination account not authorized (ERROR)
13502	Invalid service (ERROR)

8.6.2 Service Activation Synchronization

Service activation requests are subject to the standard data synchronization protocol. The scope of these requests and the <TOKEN> is the user ID. The request and response tags are <ACCTSYNCRQ> and <ACCTSYNCRS>.

8.6.3 Examples

Activating a payment:

```
<ACCTTRNRQ>
   <TRNUID>12345
   <ACCTRQ>
      <SVCADD>
         <BANKACCTTO>
            <BANKID>1234567789
            <ACCTID>12345
            <ACCTTYPE>CHECKING
         </BANKACCTTO>
      </SVCADD>
      <SVC>BPSVC
   </ACCTRQ>
</ACCTTRNRQ>
```

A response:

```
<ACCTTRNRS>
   <TRNUID>12345
   <STATUS>
      <CODE>0
      <SEVERITY>INFO
   </STATUS>
   <ACCTRS>
      <SVCADD>
         <BANKACCTTO>
            <BANKID>1234567789
            <ACCTID>12345
            <ACCTTYPE>CHECKING
         </BANKACCTTO>
      </SVCADD>
      <SVC>BPSVC
      <SVCSTATUS>ACTIVE
   </ACCTRS>
</ACCTTRNRS>
```

8.7 Name and Address Changes

Users may request that an FI update the official name, address, phone, and e-mail information using the <CHGUSERINFORQ>. All modified and unmodified elements are submitted in a change user information request, <CHGUSERINFORQ>. The lack of inclusion of a field in a change user request when that field was previously populated implies its deletion on the server. The response reports all of the current values. An optional USERID has been added to message set version 2 to facilitate the change of user info by proxy. An example of which would be a customer service representative who might be asked by the end user to change their address. If the USERID tag is not present in CHGUSERINFO, then the USERID from the SONRQ is assumed to be the identifier for the user in question. For security reasons, some of the fields in the <ENROLLRQ> cannot be changed online, such as tax ID and userID.

The transaction tag is <CHGUSERINFOTRNRQ> and <CHGUSERINFOTRNRS>. These messages are subject to synchronization, <CHGUSERINFOSYNCRQ> and <CHGUSERINFOSYNCRS>.

8.7.1 Change User Information Request <CHGUSERINFORQ>

Tag	*Version*	*Description*
<CHGUSERINFORQ>		Change-user-information-request aggregate
<USERID>	V2 only	User ID, ***A-32***
<FIRSTNAME>		First name of user, ***A-32***
<MIDDLENAME>		Middle name of user, ***A-32***
<LASTNAME>		Last name of user, ***A-32***
<ADDR1>		Address line 1, ***A-32***
<ADDR2>		Address line 2, ***A-32***
<ADDR3>		Address line 3, ***A-32***
<CITY>		City, ***A-32***
<STATE>		State or province, ***A-5***
<POSTALCODE>		Postal code, ***A-11***
<COUNTRY>		3-letter country code from ISO/DIS-3166, ***A-3***
<DAYPHONE>		Daytime telephone number, ***A-32***
<EVEPHONE>		Evening telephone number, ***A-32***
<EMAIL>		Electronic e-mail address, ***A-80***
</CHGUSERINFORQ>		

8.7.2 Change User Information Response <CHGUSERINFORS>

Tag	*Version*	*Description*
<CHGUSERINFORS>		Change-user-information-request aggregate
<USERID>	V2 only	User ID, ***A-32***
<FIRSTNAME>		First name of user, ***A-32***
<MIDDLENAME>		Middle name of user, ***A-32***
<LASTNAME>		Last name of user, ***A-32***

Tag	Version	Description
<ADDR1>		Address line 1, ***A-32***
<ADDR2>		Address line 2, ***A-32***
<ADDR3>		Address line 3, ***A-32***
<CITY>		City, ***A-32***
<STATE>		State or province, ***A-5***
<POSTALCODE>		Postal code, ***A-11***
<COUNTRY>		3-letter country code from ISO/DIS-3166, ***A-3***
<DAYPHONE>		Daytime telephone number, ***A-32***
<EVEPHONE>		Evening telephone number, ***A-32***
<EMAIL>		Electronic e-mail address, ***A-80***
<DTINFOCHG>		Date and time of update ***datetime***
</CHGUSERINFORS>		

8.7.3 Status Codes

Code	Meaning
0	Success (INFO)
2000	General error (ERROR)
13503	Cannot change user information (ERROR)

8.8 Signup Message Set Profile Information

A server must include the following aggregates as part of the profile <MSGSETLIST> response, since every server must support at least the account information and service activation messages. Servers indicate how enrollment should proceed: via the client, a given web page, or a text message directing users to some other method (such as a phone call).

Tag	Version	Description
<SIGNUPMSGSET>		Signup-message-set-profile-information aggregate
<SIGNUPMSGSETV1>		Opening tag for V1 of the message set profile information
		Same as V2, other than <PREAUTH> tag added to V2
<MSGSETCORE>		Common message set information, defined in Chapter 7, “FI Profile”
</MSGSETCORE>		
Enrollment options. Choose one of <CLIENTENROLL>, <WEBENROLL>, or <OTHERENROLL>.		
<CLIENTENROLL**>**		Client-based enrollment supported
<ACCTREQUIRED>		Y if account number is required as part of enrollment, ***Boolean***
</CLIENTENROLL**>**		
-or-		
<WEBENROLL**>**		Web-based enrollment supported
<URL>		URL to start enrollment process, ***URL***
</WEBENROLL>		
-or-		

Tag	*Version*	*Description*
<OTHERENROLL**>**		Some other enrollment process
<MESSAGE>		Message to consumer about what to do next (for example, a phone number), ***A-80***
</OTHERENROLL**>**		
<CHGUSERINFO>		Y if server supports client-based user information changes, ***Boolean***
<AVAILACCTS>		Y if server can provide information on accounts with SVCSTATUS available, N means client should expect to ask user for specific account information, ***Boolean***
<CLIENTACTREQ>		Y if server allows clients to make service activation requests (<ACCTRQ>), N if server will only advise clients via synchronization of service additions, changes, or deletions. ***Boolean***
</SIGNUPMSGSETV1>		
<SIGNUPMSGSETV2>		Opening tag for V2 of the message set profile information
		Same as V1, other than <PREAUTH> tag added to V2
<MSGSETCORE>		Common message set information, defined in Chapter 7, "FI Profile"
</MSGSETCORE>		
Enrollment options. Choose one of <CLIENTENROLL>, <WEBENROLL>, or <OTHERENROLL>.		
<CLIENTENROLL**>**		Client-based enrollment supported
<ACCTREQUIRED>		Y if account number is required as part of enrollment, ***Boolean***
</CLIENTENROLL**>**		
-or-		
<WEBENROLL**>**		Web-based enrollment supported
<URL2>		URL to start enrollment process, ***URL2***
</WEBENROLL>		
-or-		
<OTHERENROLL**>**		Some other enrollment process
<MESSAGE>		Message to consumer about what to do next (for example, a phone number), ***A-80***
</OTHERENROLL**>**		
<CHGUSERINFO>		Y if server supports client-based user information changes, ***Boolean***
<AVAILACCTS>		Y if server can provide information on accounts with SVCSTATUS available, N means client should expect to ask user for specific account information, ***Boolean***
<CLIENTACTREQ>		Y if server allows clients to make service activation requests (<ACCTRQ>), N if server will only advise clients via synchronization of service additions, changes, or deletions. ***Boolean***
<PREAUTH>		Y if server supports <PREAUTHTOKEN> for account activation, ***Boolean.***
</SIGNUPMSGSETV2>		
</SIGNUPMSGSET>		

9. Customer to FI Communication

9.1 The E-Mail Message Set

The e-mail message set includes two messages: generic e-mail and generic MIME requests by way of URLs. In Open Financial Exchange files, the message set names are EMAILMSGSV1 and EMAILMSGSV2.

9.2 E-Mail Messages

Open Financial Exchange allows consumers and FIs to exchange messages. The message body can be placed in HTML so that FIs can provide some graphic structure to the message. Keep in mind that, as with regular World Wide Web browsing, an Open Financial Exchange client might not support some or all of the HTML formatting, so the text of the message must be clear on its own. Clients can request the server to send graphics (the images referenced in an <IMG> tag) as part of the response file, or clients can separately request those elements. If a server sends images, it should use the standard procedure for incorporating external data as described in Chapter 2, "Structure." Servers are not required to support HTML or to send images, even if the client asks.

A user or an FI can originate a message. E-mail messages are subject to data synchronization so that a server can send a response again if it is lost or if multiple clients use it.

Because e-mail messages cannot be replied to immediately, the response should just echo back the original message (so that data synchronization will get this original e-mail message to other clients). When the FI is ready to reply, it should generate an unsolicited response (<TRNUID>0) and the client will pick this up during synchronization.

Client Sends	*Server Responds*
Account information	
From, To	
Subject	
Message	
	Account information
	From, To
	Subject
	Message
	Type

9.2.1 Regular vs. Specialized E-Mail

Several services with Open Financial Exchange define e-mail requests and responses that contain additional information specific to that service. To simplify implementation for clients and servers, this section defines a <MAIL> aggregate that Open Financial Exchange uses in all e-mail requests and responses. For regular e-mail, the only additional information is an account-from aggregate and whether to include images in the e-mail response or not.

9.2.2 Basic <MAIL> Aggregate

Tag	*Description*
<MAIL>	Core e-mail aggregate

Tag	Description
<USERID>	User ID such as SSN, ***A-32***
<DTCREATED>	When message was created, ***datetime***
<FROM>	Who the message is from, ***A-32***
<TO>	Who the message should be delivered to, ***A-32***
<SUBJECT>	Subject of message (plain text, not HTML), ***A-60***
<MSGBODY>	Body of message, HTML-encoded or plain text depending on <USEHTML>,
	HTML-encoded text = ***A-10000***
	Plain text = ***A-2000***
</MSGBODY>	End of message
<INCIMAGES>	Include images in the message body. ***Boolean***
<USEHTML>	Y for HTML-formatted text. N for plain text. See section 9.2.2.2 for more information. ***Boolean***
</MAIL>	

9.2.2.1 <INCIMAGES>

The meaning of the <INCIMAGES> tag depends on whether the tag appears in a request or response.

When used in a request, <INCIMAGES> indicates whether the client accepts mail that includes images in the message body.

When used in a request...	Description
<INCIMAGES>Y	The client accepts mail that includes images in the message body. In this case, the server can choose whether to send images in the response.
<INCIMAGES>N	The client does not accept mail that includes images in the message body. In this case, the server must not send images in the response.

When used in a response, <INCIMAGES> indicates whether the server included images in the message body.

When used in a response…	Description
<INCIMAGES>Y	The server included images in the message body.
<INCIMAGES>N	The server did not include images in the message body.

9.2.2.2 <USEHTML>

The meaning of the <USEHTML> tag depends on whether the tag appears in a request or response.

When used in a request, <USEHTML> indicates whether the client sends and accepts HTML-formatted text in the message body.

When used in a request…	Description
<USEHTML>Y	The client is including HTML-formatted text in the message body. In addition, the client will accept mail responses that include HTML-formatted text in the message body. In this case, a server can choose whether to respond with HTML-formatted text or plain text.
<USEHTML>N	The client is not including HTML-formatted text in the message body. In addition the client will not accept mail responses that include HTML-formatted text in the message body.

When used in a response, <USEHTML> indicates whether the message body includes HTML-formatted text or plain text.

When used in a response…	Description

When used in a response…	Description
<USEHTML>Y	The server is including HTML-formatted text in the message body.
<USEHTML>N	The server is including only plain text in the message body.

Note: *When using HTML for the message body, clients and servers are REQUIRED to enclose the HTML in an SGML-marked section to protect the HTML markup: <![CDATA [... html ...]]>. For an example, see section 9.2.5.*

9.2.3 E-Mail <MAILRQ> <MAILRS>

E-mail is subject to synchronization. The transaction tag is <MAILTRNRQ> / <MAILTRNRS> and the synchronization tag is <MAILSYNCRQ> / <MAILSYNCRS>.

Tag	Description
<MAILRQ>	E-mail-message-request aggregate
<MAIL>	Core e-mail aggregate
</MAIL>	
</MAILRQ>	

In a response, the <TRNUID> is zero if this is an unsolicited message. Otherwise, it should contain the <TRNUID> of the user's original message. It is RECOMMENDED that servers include the <MESSAGE> of the user's message as part of the reply <MESSAGE>. The <MESSAGE> contents can include carriage returns to identify desired line breaks.

Tag	Description
<MAILRS>	E-mail-message-response aggregate
<MAIL>	Core e-mail aggregate
</MAIL>	
</MAILRS>	

9.2.3.1 Status Codes

Code	Meaning
0	Success (INFO)
2000	General error (ERROR)
16500	HTML not allowed (ERROR)
16501	Unknown mail To: (ERROR)

9.2.4 E-Mail Synchronization <MAILSYNCRQ> <MAILSYNCRS>

E-mail presents a special case with regards to synchronization. Since FIs will not immediately reply to a user's e-mail, the response to the user's e-mail only echoes the request and confirms that the e-mail was successfully received. The client receives the real response to the e-mail following a synchronization request.

Note that this synchronization action expects only the basic <MAILRS> responses. Specialized e-mail is received by means of their own synchronization requests.

Tag	Version	Description

Tag	Version	Description
<MAILSYNCRQ>		E-mail-synchronization-request aggregate
Client synchronization option; <TOKEN>, <TOKEN2>, <TOKENONLY>, or <REFRESH>		
<TOKEN>	V1 only	Previous value of <TOKEN> received for this type of synchronization request from server; 0 for first-time requests; ***token***
<TOKEN2>	V2 only	Previous value of <TOKEN2> received for this type of synchronization request from server; 0 for first-time requests; ***token2***
<TOKENONLY>		Request for just the current <TOKEN> without the history, ***Boolean***
<REFRESH>		Request for refresh of current state, ***Boolean***
<REJECTIFMISSING>		If Y, do not process requests if client <TOKEN> is out of date, ***Boolean***
<INCIMAGES>		Y if the client accepts mail with images in the message body, N if the client does not accept mail with images in the message body, ***Boolean***
<USEHTML>		Y if client wants an HTML response, N if client wants plain text, ***Boolean***
<MAILTRNRQ>		Mail-transaction-request aggregate (0 or more)
</MAILTRNRQ>		
</MAILSYNCRQ>		

Tag	Version	Description
<MAILSYNCRS>		E-mail-synchronization-response. aggregate
<TOKEN>	V1 only	Server history marker, ***token***
<TOKEN2>	V2 only	Server history marker, ***token2***
<LOSTSYNC>		Y if the token in the synchronization request is older than the earliest entry in the server's history table. In this case, some responses have been lost. N if the token in the synchronization request is newer than or matches a token in the server's history table. ***Boolean***
<SYNCERROR>	V2 only	Optional error code, ***N-6***
<MAILTRNRS>		Missing e-mail response transactions (0 or more)
</MAILTRNRS>		
</MAILSYNCRS>		

9.2.5 E-Mail Example

In this example, a consumer requests information about the checking statement just downloaded. Since the financial institution will not immediately answer the inquiry, the immediate response only echoes the consumer's request and confirms that the request was successfully received.

The client receives the real response at a later time following a mail synchronization request. For an example of the mail synchronization request and response, see section 9.2.5.1.

> ***Note:*** *This example omits the <OFX> top level and the signon <SONRQ>. Since this example uses HTML for the message body, it must protect the HTML content in an SGML CDATA-marked section.*

The request:

```
<MAILTRNRQ>
   <TRNUID>54321
   <MAILRQ>
```

```
      <MAIL>
         <USERID>123456789
         <FROM>James Hackleman
         <TO>Noelani Federal Savings
         <SUBJECT>What do I need to earn interest?
         <DTCREATED>19960305
         <MSGBODY><![ CDATA [<HTML><BODY>I didn't earn any interest this month. Can
you please tell me what I need to do to earn interest on this account?</BODY></HTML>
]]></MSGBODY>
         <INCIMAGES>N
         <USEHTML>Y
      </MAIL>
   </MAILRQ>
</MAILTRNRQ>
```

The response from the FI:

```
<MAILTRNRS>
   <TRNUID>54321
   <STATUS>
      <CODE>0
      <SEVERITY>INFO
   </STATUS>
   <MAILRS>
      <MAIL>
         <USERID>123456789
         <FROM>James Hackleman
         <TO>Noelani Federal Savings
         <SUBJECT>What do I need to earn interest?
         <DTCREATED>19960305
         <MSGBODY><![ CDATA [<HTML><BODY>I didn't earn any interest this month. Can
you please tell me what I need to do to earn interest on this account?</BODY></HTML>
]]></MSGBODY>
         <INCIMAGES>N
         <USEHTML>Y
      </MAIL>
   </MAILRS>
</MAILTRNRS>
```

9.2.5.1 E-Mail Synchronization Example

In the following example, the client has not yet received the reply to the e-mail sent in the previous example, so its <TOKEN> is one less than the server's. The server replies by giving the current <TOKEN> and the missed response.

```
<MAILSYNCRQ>
   <TOKEN>101
   <REJECTIFMISSING>N
   <INCIMAGES>N
   <USEHTML>Y
</MAILSYNCRQ>

<MAILSYNCRS>
   <TOKEN>102
   <MAILTRNRS>
      <TRNUID>0                          <!-- server initiated response →
      <STATUS>
         <CODE>0
         <SEVERITY>INFO
      </STATUS>
      <MAILRS>
         <MAIL>
            <USERID>123456789
            <DTCREATED>19960307
            <FROM>Noelani Federal Savings
```

```
            <TO>James Hackleman
            <SUBJECT>Re: What do I need to earn interest?
            <MSGBODY>><![ CDATA [<HTML><BODY>You need to maintain $1000 in this
account to earn interest. Because your balance was only $750 this month, no interest
was earned. You could also switch to our new Checking Extra plan that always pays
interest. Call us or check our web page http://www.fi.com/check-plans.html for more
information.
Sincerely,
Customer Service Department

Original message:
I didn't earn any interest this month. Can you please tell me what I need to do to
earn interest on this account?</BODY></HTML>
]]></MSGBODY>
            <INCIMAGES>N
            <USEHTML>Y
         </MAIL>
      </MAILRS>
   </MAILTRNRS>
</MAILSYNCRS>
```

9.3 Get HTML Page

Some responses contain values that are URLs intended to be separately fetched by clients. Clients can use their own HTTP libraries to perform this fetch outside of the Open Financial Exchange specification. However, to insulate clients against changes in transport technology, and to allow for fetches that require the protection of an authenticated signon by a specific user, Open Financial Exchange defines a transaction roughly equivalent to an HTTP Get. Any MIME type can be retrieved, including images as well as HTML pages.

9.3.1 MIME Get Request and Response <GETMIMERQ> <GETMIMERS>

The following table lists the components of a request:

Tag	*Version*	*Description*
<GETMIMERQ>		Get-MIME-request aggregate
<URL>	V1 only	URL, ***URL***
<URL2>	V2 only	URL, ***URL2***
</GETMIMERQ>		

The response simply echoes the URL. The actual response, whether HTML, an image, or some other type, is always sent as a separate part of the file using multi-part MIME.

Tag	*Version*	*Description*
<GETMIMERS>		Get-MIME-response aggregate
<URL>	V1 only	URL, ***URL***
<URL2>	V2 only	URL, ***URL2***
</GETMIMERS>		

9.3.1.1 Status Codes

Code	*Meaning*
0	Success (INFO)
2000	General error (ERROR)

Code	Meaning
2019	Duplicate request (ERROR)
16502	Invalid URL (ERROR)
16503	Unable to get URL (ERROR)

9.3.2 MIME Example

A request:

```
<GETMIMETRNRQ>
   <TRNUID>54321
   <GETMIMERQ>
      <URL>http://www.fi.com/apage.html
   </GETMIMERQ>
</GETMIMETRNRQ>
```

A response – the full file is shown here to illustrate the use of multi-part MIME:

```
HTTP 1.0 200 OK
Content-Type: multipart/x-mixed-replace; boundary =--boundary—

--boundary--
Content-Type: application/x-ofx
Content-Length: 8732

OFXHEADER:100
DATA:OFXSGML
VERSION:102
SECURITY:TYPE1
ENCODING:USASCII

<OFX>
      <!-- signon not shown
      message set wrappers not shown -->
<GETMIMETRNRS>
   <TRNUID>54321
   <STATUS>
      <CODE>0
      <SEVERITY>INFO
   </STATUS>
   <GETMIMERS>
      <URL>http://www.fi.com/apage.html
   </GETMIMERS>
</GETMIMETRNRS>
</OFX>

--boundary--
Content-Type: text/html
<HTML>
   <!-- standard HTML page -->
</HTML>

--boundary--
```

9.4 E-Mail Message Set Profile Information

If either or both of the messages in the e-mail message set are supported, the following aggregate must be included in the profile <MSGSETLIST> response.

Tag	Description

Tag	*Description*
<EMAILMSGSET>	E-mail-message-set-profile-information aggregate
<EMAILMSGSETV1>	Opening tag for V1 of the message set profile information
<MSGSETCORE>	Common message set information, defined in Chapter 7, “FI Profile”
</MSGSETCORE>	
<MAILSUP>	Y if server supports generic e-mail message, ***Boolean***
<GETMIMESUP>	Y if server supports get MIME message, ***Boolean***
</EMAILMSGSETV1>	
<EMAILMSGSETV2>	Opening tag for V2 of the message set profile information
<MSGSETCORE>	Common message set information, defined in Chapter 7, “FI Profile”
</MSGSETCORE>	
<MAILSUP>	Y if server supports generic e-mail message, ***Boolean***
<GETMIMESUP>	Y if server supports get MIME message, ***Boolean***
</EMAILMSGSETV2>	
</EMAILMSGSET>	

10. Recurring Transactions

Open Financial Exchange enables users to automate transactions that occur on a regular basis. Recurring transactions are useful when a customer has payments or transfers, for example, that repeat at regular intervals. The customer can create a "model" at the server for automatic generation of these instructions. The model in turn creates payments or transfers until it is canceled or expires. After the user creates a recurring model at the server, the server can relieve the user from the burden of creating these transactions; it generates the transactions on its own, based on the operating parameters of the model.

10.1 Creating a Recurring Model

The client must provide the following information to create a model:

- Type of transaction generated by the model (payment or transfer)
- Frequency of recurring transaction
- Total number of recurring transactions to generate
- Service-specific information, such as transfer date, payment amount, payee address

The model creates each transaction some time before its due date, usually thirty days. This allows the user to retrieve the transactions in advance of posting. This also gives the user the opportunity to modify or cancel individual transactions without changing the recurring model itself.

When a model is created, it can generate several transactions immediately. The model does not automatically return responses for the newly created transactions. It returns a response only to the request that was made to create the model. For this reason, clients should send a synchronization request along with the request to create a model. This allows the server to return the newly created transaction responses, as well as the response to the request to set up a new model.

10.2 Recurring Instructions <RECURRINST>

The Recurring Instructions aggregate is used to specify the schedule for a repeating instruction. It is passed to the server when a recurring transfer or payment model is first created.

Tag	*Description*
<RECURRINST>	Recurring-Instructions aggregate
<NINSTS>	Number of instructions If this tag is absent, the schedule is open-ended, ***N-3***
<FREQ>	Frequency, see section 10.2.1
</RECURRINST>	

10.2.1 Values for <FREQ>

Value	*Description*
WEEKLY	Weekly
BIWEEKLY	Biweekly
TWICEMONTHLY	Twice a month
MONTHLY	Monthly
FOURWEEKS	Every four weeks
BIMONTHLY	Bimonthly
QUARTERLY	Quarterly

Value	Description
SEMIANNUALLY	Semiannually
ANNUALLY	Annually

Rules for calculating recurring dates of WEEKLY, BIWEEKLY, and TWICEMONTHLY are as follows:

- WEEKLY = starting date for first transaction, starting date + 7 days for the second
- TWICEMONTHLY = starting date for first, starting date + 15 days for the second
- BIWEEKLY = starting date for first, starting date + 14 days for the second

Examples:

Start date of May 2: next transaction date for WEEKLY is May 9; TWICEMONTHLY is May 17; next transfer date for BIWEEKLY is May 16.

Start date of May 20: next date for WEEKLY is May 27; TWICEMONTHLY is June 4; next date for BIWEEKLY is June 3.

TWICEMONTHLY recurring transactions will occur each month on those days adjusting for weekends and holidays. BIWEEKLY will occur every 14 days.

10.2.2 Examples

The following example illustrates the creation of a repeating payment. The payment repeats on a monthly basis for 12 months. All payments are for $395.

The request:

```
.
.
.
<RECPMTRQ>
   <RECURRINST>
      <NINSTS>12
      <FREQ>MONTHLY
   </RECURRINST>
   <PMTINFO>
      <BANKACCTFROM>
         <BANKID>555432180
         <ACCTID>763984
         <ACCTTYPE>CHECKING
      </BANKACCTFROM>
      <TRNAMT>395.00
      <PAYEEID>77810
      <PAYACCT>444-78-97572
      <DTDUE>19971115
      <MEMO>Auto loan payment
   </PMTINFO>
</RECPMTRQ>
.
.
.
```

The response includes the <RECSRVRTID> that the client can use to cancel or modify the model:

```
.
.
.
<RECPMTRS>
   <RECSRVRTID>387687138
   <RECURRINST>
```

```
      <NINSTS>12
      <FREQ>MONTHLY

   </RECURRINST>
   <PMTINFO>
      <BANKACCTFROM>
         <BANKID>555432180
         <ACCTID>763984
         <ACCTTYPE>CHECKING
      </BANKACCTFROM>
      <TRNAMT>395.00
      <PAYEEID>77810
      <PAYACCT>444-78-97572
      <DTDUE>19971115
      <MEMO>Auto loan payment
   </PMTINFO>
</RECPMTRS>
.
.
.
```

10.3 Retrieving Transactions Generated by a Recurring Model

Once created, a recurring model independently generates instructions. Since the client has not directly generated these transactions, the client has no record of their creation. To enable users to modify and/or cancel pending instructions, the client must use data synchronization in order to retrieve these transactions.

The client has two purposes for synchronizing state with the server with respect to recurring models:

- Retrieve any added, modified, or canceled recurring models
- Retrieve any added, modified, or canceled transactions generated by any models

The client must be able to synchronize with the state of any models at the server, as well as the state of any transactions generated by the server.

10.4 Modifying and Canceling Individual Transactions

Once created and retrieved by the customer, recurring payments and transfers are almost identical to customer-created payments or transfers. As with ordinary payments or transfers, you can cancel or modify transactions individually. However, because servers generate these transfers, they are different in the following respects:

- Recurring transactions must be retrieved as part of a synchronization request.
- Recurring transactions are related to a model. A server can modify or cancel transactions if the model is modified or canceled.

10.5 Modifying and Canceling Recurring Models

A recurring model can be modified or canceled. When a model is modified, all transactions that it generates in the future will change as well. The client can indicate whether transactions that have been generated, but have not been sent, should be modified as well. The actual elements within a transaction that can be modified differ by service. See the recurring sections within Chapter 11, “Banking,” and Chapter 12, “Payments,” for details.

A user can cancel a model immediately or at a future date. If a user cancels the model immediately, the client cancels any transactions that it has not yet sent. If the client schedules the cancel for a future date, the client will not cancel pending transactions.

10.5.1 Examples

Canceling a recurring payment model requires the client to pass the <RECSRVRTID> of the model. The client requests that pending payments also be canceled. The server cancels the model immediately and notifies the client that both the model and any scheduled payments were canceled.

The request:

```
.
.
.
   <RECPMTCANCRQ>
      <RECSRVRTID>387687138
      <CANPENDING>Y
   </RECPMTCANCRQ>
.
.
.
```

The response:

```
.
.
.
   <RECPMTCANCRS>
      <RECSRVRTID>387687138
      <CANPENDING>Y
   </RECPMTCANCRS>
.
.
.
```

The server also cancels any payments that have been generated but not executed. In the example shown above, the client would not learn of this immediately. To receive notification that the model and all generated payments were canceled, the client would need to include a synchronization request in the file. The following example illustrates this alternate approach.

The request file now includes a synchronization request:

```
.
.
.
   <RECPMTCANCRQ>
      <RECSRVRTID>387687138
      <CANPENDING>Y
   </RECPMTCANCRQ>
   <PMTSYNCRQ>
      <TOKEN>12345
      <REJECTIFMISSING>N
      <BANKACCTFROM>
         <BANKID>123432123
         <ACCTID>516273
         <ACCTTYPE>CHECKING
```

```
      </BANKACCTFROM>
   </PMTSYNCRQ>
.
.
.
```

The response file now contains two responses (assuming one payment was pending), one for the canceled model and one for the canceled payment.

```
.
.
.
   <RECPMTCANCRS>
      <RECSRVRTID>387687138
      <CANPENDING>Y
   </RECPMTCANCRS>
   <PMTSYNCRS>
      <TOKEN>3247989384
      <BANKACCTFROM>
         <BANKID>123432123
         <ACCTID>516273
         <ACCTTYPE>CHECKING
      </BANKACCTFROM>
      <PMTTRNRS>
         <TRNUID>10103
         <STATUS>
            <CODE>0
            <SEVERITY>INFO
         </STATUS>
         <PMTCANCRS>
            <SRVRTID>1030155
         </PMTCANCRS>
      </PMTTRNRS>
   </PMTSYNCRS>
.
.
.
```

11. Banking

Open Financial Exchange enables financial institution (FI) customers to keep their finances up-to-date and to manage their bank accounts conveniently in several ways. Customers can download transactions and update account balances on a daily basis. They can retrieve a closing statement that contains the same information that they are accustomed to seeing on a paper statement. They can transfer funds between accounts at a financial institution, either immediately upon going online or on a regular schedule. Customers can schedule transfers between accounts on a recurring basis and can transfer funds between accounts at different financial institutions. If necessary, customers can request a wire funds transfer. Open Financial Exchange also enables requests to stop payment on pending checks.

Using customer notification, an FI can notify customers of important events regarding their accounts, such as returned checks or deposits.

11.1 Consumer and Business Banking

Open Financial Exchange supports banking for both consumers and businesses. Some customers might use some areas more heavily within Open Financial Exchange Banking (such as credit card download); other areas might be more appropriate for businesses (such as wire transfers). Yet all of the functionality defined for Banking is appropriate to some extent for both consumer and business applications.

11.2 Credit Card Data

Credit card data is available to Open Financial Exchange clients through the statement download facility. Statement download provides a way to download credit card transaction data and balances on an as-needed basis. Statement closing information can be made available to clients as well.

11.3 Common Banking Aggregates

This section describes several aggregates used throughout the Banking portion of Open Financial Exchange.

11.3.1 Banking Account <BANKACCTFROM> and <BANKACCTTO>

Open Financial Exchange uses the Banking Account aggregates to identify an account at an FI. The aggregates contain enough information to uniquely identify an account for the purposes of statement download, bill payment, and funds transfer. <CCACCTFROM> identifies credit card accounts; see section 11.3.2.

Tag	*Version*	*Description*
<BANKACCTFROM>		Bank-account-from aggregate
<BANKID>		Bank identifier, ***A-9***
		Use of this field by country:
		COUNTRY Interpretation
		BEL Bank code
		CAN Routing and transit number
		CHE Clearing number
		DEU ***Bankleitzahl***
		ESP ***Entidad***
		FRA ***Banque***

Tag	Version	Description
		GBR Sort code
		ITA ***ABI***
		NLD Not used (field contents ignored)
		USA Routing and transit number
<BRANCHID>		Branch identifier. May be required for some non-US banks, ***A-22***
		Use of this field by country:
		COUNTRY Interpretation
		BEL Not present
		CAN Not present
		CHE Not present
		DEU Not present
		ESP ***Oficina***
		FRA ***Agence***
		GBR Not present
		ITA ***CAB***
		NLD Not present
		USA Not present
<ACCTID>		Account number, ***A-22***
<ACCTTYPE>		Type of account, see section 11.3.1.2
<ACCTKEY>		Checksum, ***A-22***
		Use of this field by country:
		COUNTRY Interpretation
		BEL Check digits
		CAN Not present
		CHE Not present
		DEU Not present
		ESP ***D.C.***
		FRA ***Clé***
		GBR Not present
		ITA ***CIN***
		NLD Not present
		USA Not present
</BANKACCTFROM>		

Tag	Version	Description
<BANKACCTTO>		Bank-account-to aggregate
<BANKID>		Bank identifier, ***A-9***
		Use of this field by country:
		COUNTRY Interpretation
		BEL Bank code
		CAN Routing and transit number

Tag	Version	Description	
		CHE	Clearing number
		DEU	***Bankleitzahl***
		ESP	***Entidad***
		FRA	***Banque***
		GBR	Sort code
		ITA	***ABI***
		NLD	Not used (field contents ignored)
		USA	Routing and transit number
<BRANCHID>		Branch identifier. May be required for some banks, ***A-22***	
		Use of this field by country:	
		COUNTRY	Interpretation
		BEL	Not present
		CAN	Not present
		CHE	Not present
		DEU	Not present
		ESP	***Oficina***
		FRA	***Agence***
		GBR	Not present
		ITA	***CAB***
		NLD	Not present
		USA	Not present
<ACCTID>		Account number, ***A-22***	
<ACCTTYPE>	V1 only	Type of account, see section 11.3.1.2	
<ACCTTYPE2>	V2 only	Type of account, see section 11.3.1.2	
<ACCTKEY>		Checksum, ***A-22***	
		Use of this field by country:	
		COUNTRY	Interpretation
		BEL	Check digits
		CAN	Not present
		CHE	Not present
		DEU	Not present
		ESP	***D.C.***
		FRA	***Clé***
		GBR	Not present
		ITA	***CIN***
		NLD	Not present
		USA	Not present
<EXTBANKACCTTO>	V2 only	Extended bank account-to information aggregate. This aggregate is only present when <BANKACCTTO> appears within the Bil Payment message set. See section 11.3.1.1.	
		Presence of this field by country:	
		COUNTRY	Interpretation

Tag	Version	Description	
		BEL	Not present
		CAN	Not present
		CHE	Required for payments and payees
		DEU	Required for payments and payees
		ESP	Not present
		FRA	Not present
		GBR	Not present
		ITA	Required for payments and payees
		NLD	Not present
		USA	Not present
</EXTBANKACCTTO>	V2 only		
</BANKACCTTO>			

11.3.1.1 Extended Banking Account Information <EXTBANKACCTTO>

The <EXTBANKACCTTO> aggregate contains additional information used to identify an account being used for a payment transfer and payee requests. It only appears within <BANKACCTTO> aggregates that are used within the Bill Payment message set, and not inside any version 1 message sets. The <EXTBANKACCTTO> aggregate is only used ifCOUNTRY is CHE, DEU or ITA.

Tag	Version	Description
<EXTBANKACCTTO>	V2 only	Extended Bank-account-to aggregate
<BANKNAME>	V2 only	Bank name, A-32 (mandatory if COUNTRY = CHE, DEU or ITA)
<BANKBRANCH>	V2 only	Bank branch name, A-32 (mandatory if COUNTRY = ITA)
<BANKCITY>	V2 only	Bank branch city, A-32 (mandatory if COUNTRY = CHE)
<BANKPOSTALCODE>	V2 only	Bank branch postal code, A-11 (mandatory if COUNTRY = CHE)
<CHE.PTTACCTID>	V2 only	PTT account number for indirect payments, A-9 (optional if COUNTRY = CHE)
</EXTBANKACCTTO>	V2 only	

11.3.1.2 Account Types for <ACCTTYPE> and <ACCTTYPE2> Elements

Type	Version	Description
CHECKING		Checking
SAVINGS		Savings
MONEYMRKT		Money Market
CREDITLINE		Line of credit
CMA	<ACCTTYPE2> only	Cash Management Account

11.3.2 Credit Card Account <CCACCTFROM>

Open Financial Exchange uses the Credit Card Account aggregate to identify a credit card account at an FI. The aggregate contains enough information to uniquely identify an account for the purposes of statement downloads and funds transfer. It is not necessary to support the Credit Card Message Set in order to use the Credit card account aggregate.

Tag	*Description*
<CCACCTFROM>	Credit-card-account-from aggregate
<ACCTID>	Account number, ***A-22***
<ACCTKEY>	Checksum for international banks, ***A-22***
</CCACCTFROM>	

The <CCACCTTO> aggregate contains the same elements.

11.3.3 Bank Account Information <BANKACCTINFO>

Open Financial Exchange uses the bank account information aggregate to download account information from an FI. It includes account number specification in <BANKACCTFROM> as well as the status of the service.

Tag	*Description*
<BANKACCTINFO>	Bank-account-information aggregate
<BANKACCTFROM>	Bank-account-from aggregate
</BANKACCTFROM>	
<SUPTXDL>	Y if account supports transaction detail downloads, N if it is balance-only, ***Boolean***
<XFERSRC>	Y if account is enabled as a source for an intrabank or interbank transfer, ***Boolean***
<XFERDEST>	Y if account is enabled as a destination for an intrabank or interbank transfer, ***Boolean***
<SVCSTATUS>	Status of the account
	AVAIL = Available, but not yet requested
	PEND = Requested, but not yet available
	ACTIVE = In use
</BANKACCTINFO>	

11.3.4 Credit Card Account Information <CCACCTINFO>

Open Financial Exchange uses the credit card account information aggregate to download account information from an FI. It includes credit card number specification in <CCACCTFROM> as well as the status of the service.

Tag	*Description*
<CCACCTINFO>	Credit-card-account-information aggregate
<CCACCTFROM>	Credit-card-account-from aggregate
</CCACCTFROM>	

Tag	Description
<SUPTXDL>	Y if account supports transaction detail downloads, N if it is balance-only, ***Boolean***
<XFERSRC>	Y if account is enabled as a source for an intrabank or interbank transfer, ***Boolean***
<XFERDEST>	Y if account is enabled as a destination for an intrabank or interbank transfer, ***Boolean***
<SVCSTATUS>	Status of the account AVAIL = Available, but not yet requested PEND = Requested, but not yet available ACTIVE = In use
</CCACCTINFO>	

11.3.5 Transfer Information <XFERINFO>

Many of the transfer requests and responses use an <XFERINFO> aggregate. This aggregate identifies accounts that are part of the transfer, amount of money to be transferred, and the date of the transfer.

Tag	Version	Description
<XFERINFO>		Transfer-information aggregate
Account-from options. Choose either <BANKACCTFROM> or <CCACCTFROM>.		
<BANKACCTFROM>		Account-from aggregate, see section 11.3.1
</BANKACCTFROM>		
-or-		
<CCACCTFROM>		Credit-card-account-from aggregate, see section 11.3.2
</CCACCTFROM>		
Account-to options. Choose either <BANKACCTTO> or <CCACCTTO>.		
<BANKACCTTO>		Account-to aggregate, see section 11.3.1
</BANKACCTTO>		
-or-		
<CCACCTTO>		Credit-card-account-to aggregate, see section 11.3.2
</CCACCTTO>		
<TRNAMT>		Amount of the transfer, ***amount*** This amount should be specified as a positive number.
<DTDUE>		Date that the transfer is to be sent. If the client does not specify <DTDUE>, the transfer occurs as soon as possible. <DTDUE> is required for scheduled or repeating transfers, ***datetime***
<DTAVAIL>	V2 only	The value date when the funds must be available in BANKACCTTO. This can potentially be set independently of DTDUE, ***datetime*** Note that for some countries (notably Italy), DTAVAIL can be a date in the past. This field is only supported by the server if the SUPPORTDTAVAIL tag of the Intrabank transfer profile is true (see section 11.13.2.2).

Tag	Version	Description
<MEMO2>	V2 only	This tag is not used in the US. Associated text content for the transfer.
</XFERINFO>		

11.3.6 Transfer Processing Status <XFERPRCSTS>

The Transfer Processing Status aggregate contains the current processing status for a transfer. The interpretation of the date value depends on the value of <XFERPRCCODE>.

Tag	Description
<XFERPRCSTS>	Transfer processing status aggregate
<XFERPRCCODE>	See section 11.3.6.1
<DTXFERPRC>	Transfer processing date; value depends on <XFERPRCCODE>
</XFERPRCSTS>	

11.3.6.1 Transfer Processing Status Values <XFERPRCCODE>

Value	Description
WILLPROCESSON	Will be processed on <DTXFERPRC>
POSTEDON	Posted on <DTXFERPRC>
NOFUNDSON	Funds not available to make transfer on <DTXFERPRC>
CANCELEDON	User canceled payment on <DTXFERPRC>
FAILEDON	Unable to make transfer for unspecified reasons on <DTXFERPRC>

11.4 Downloading Transactions and Balances

Statement download allows a customer to receive transactions and balances that are typically part of a regular paper statement. Clients can retrieve transactions and balances on a daily basis if they wish. Coupled with the information returned by statement closing information request (see section 11.5), a client can construct an "electronic statement" that contains all of the information that appears on a regular paper statement.

Clients typically allow customers to view these transactions and guide customers through a process of updating their account registers based on the downloaded transactions. By using transaction IDs supplied by financial institutions, Open Financial Exchange makes it possible for clients to ensure that a server downloads each transaction only once. The request also contains starting and ending dates to limit the amount of downloaded data. Clients can remember the last date they received data and use it as the starting date in the next request.

The messages in this chapter are appropriate for checking, savings, money market, credit card, and line of credit accounts. Investment statement download is a superset of bank statement download. Chapter 13, "Investments," describes the messages specific to investment statement download.

Statement download requires the client to designate an account for the download, and to indicate if the server should download transactions and/or balances. If the client wishes to download transactions, it can specify a date range that the transactions fall within.

The server returns transactions that match the date range (if the client specifies one), and balance information for the account.

Client Sends	*Server Responds*
Account information	
Include transactions?	
Date range	
	Transactions
	Cycle-ending information

11.4.1 Bank Statement Download

A client can request a download of balances separately from transaction detail. The server downloads transactions only if the <INCTRAN> aggregate is present and the <INCLUDE> flag is set to Y. The current ledger balance (and balance date) are always downloaded.

You can use the <STMTRQ> … <STMTRS> request and response pair to download transactions and balances for checking, savings, money market, and line of credit accounts. Section 11.4.2 describes download for credit card accounts.

Clients and servers should interpret <DTSTART> and <DTEND> as described in Chapter 3, "Common Aggregates, Elements, and Data Types."

11.4.1.1 Request <STMTRQ>

The <STMTRQ> request must appear within a <STMTTRNRQ> transaction wrapper.

Tag	*Description*
<STMTRQ>	Statement-request aggregate
<BANKACCTFROM>	Bank-account-from aggregate, see section 11.3.1

Tag	Description
</BANKACCTFROM>	
<INCTRAN>	Include-transactions aggregate
<DTSTART>	Start date of statement requested, ***datetime***
<DTEND>	End date of statement requested, ***datetime***
<INCLUDE>	Include transactions flag, ***Boolean***
</INCTRAN>	
</STMTRQ>	

11.4.1.2 Response <STMTRS>

A statement response comprises tags supplying various balances, plus zero or more <STMTTRN> aggregates, each describing one statement transaction.

The <STMTRS> response must appear within a <STMTTRNRS> transaction wrapper.

See Chapter 3, "Common Aggregates, Elements, and Data Types," for size and type information for common elements (such as currency values).

Tag	Description
<STMTRS>	Statement-response aggregate
<CURDEF>	Default currency for the statement, ***currsymbol***
<BANKACCTFROM>	Account-from aggregate, see section 11.3.1
</BANKACCTFROM>	
<BANKTRANLIST>	Statement-transaction-data aggregate
<DTSTART>	Start date for transaction data, ***date***
<DTEND>	Value that client should send in next <DTSTART> request to ensure that it does not miss any transactions, ***date***
<STMTTRN>	Opening tag for each statement transaction (0 or more), see section 11.4.3
</STMTTRN>	End tag for each statement transaction
</BANKTRANLIST>	
<LEDGERBAL>	Ledger balance aggregate
<BALAMT>	Ledger balance amount, ***amount***
<DTASOF>	Balance date, ***datetime***
</LEDGERBAL>	
<AVAILBAL>	Available balance aggregate
<BALAMT>	Available balance amount, ***amount***
<DTASOF>	Balance date, ***datetime***
</AVAILBAL>	
<MKTGINFO>	Marketing information (at most 1), ***A-360***
</STMTRS>	

11.4.1.3 Status Codes

Code	Meaning
0	Success
2000	General error (ERROR)
2002	General account error (ERROR)
2003	Account not found (ERROR)
2004	Account closed (ERROR)
2005	Account not authorized (ERROR)
2019	Duplicate request (ERROR)

11.4.2 Credit Card Statement Download

The credit card download request is semantically identical to the bank statement download request. However, the <CCSTMTRQ> aggregate contains the credit card request, not the <STMTRQ> aggregate.

11.4.2.1 Request <CCSTMTRQ>

The <CCSTMTRQ> request must appear within a <CCSTMTTRNRQ> transaction wrapper.

Tag	Description
<CCSTMTRQ>	Credit-card-download-request aggregate
<CCACCTFROM>	Credit-card-account-from aggregate
<ACCTID>	Account number, ***A-22***
<ACCTKEY>	Checksum for international banks, ***A-22***
</CCACCTFROM>	
<INCTRAN>	Include transactions
<DTSTART>	Start date of statement requested, ***datetime***
<DTEND>	Ending date of statement requested, ***datetime***
<INCLUDE>	Include transactions flag, ***Boolean***
</INCTRAN>	
</CCSTMTRQ>	

11.4.2.2 Response <CCSTMTRS>

The credit card download response is semantically identical to the bank statement download response. However, the <CCSTMTRS> aggregate contains the credit card response, not the <STMTRS> aggregate.

The <CCSTMTRS> response must appear within a <CCSTMTTRNRS> transaction wrapper.

Tag	Description
<CCSTMTRS>	Credit-card-download-response aggregate
<CURDEF>	Default currency for the statement, ***currsymbol***
<CCACCTFROM>	Account from aggregate, see section 11.3.2
</CCACCTFROM>	

Tag	Description
<BANKTRANLIST>	Opening tag for statement transaction data
<DTSTART>	Start date for transaction data, ***date***
<DTEND>	Value client should send in next <DTSTART> request to ensure that it does not miss any transactions, ***date***
<STMTTRN>	Opening tag for each statement transaction (0 or more), see section 11.4.3
</STMTTRN>	End tag for each statement transaction
</BANKTRANLIST>	
<LEDGERBAL>	Ledger-balance aggregate
<BALAMT>	Ledger balance amount, ***amount***
<DTASOF>	Balance date, ***datetime***
</LEDGERBAL>	
<AVAILBAL>	Available balance aggregate
<BALAMT>	Available balance amount, ***amount***
<DTASOF>	Balance date, ***datetime***
</AVAILBAL>	
<MKTGINFO>	Marketing information (at most 1), ***A-360***
</CCSTMTRS>	

11.4.2.3 Status Codes

Code	Meaning
0	Success
2001	Invalid account (ERROR)
2002	General account error (ERROR)
2003	Account not found (ERROR)
2004	Account closed (ERROR)
2005	Account not authorized (ERROR)
2019	Duplicate request (ERROR)

11.4.3 Statement Transaction <STMTTRN>

A <STMTTRN> aggregate describes a single transaction. It identifies the type of the transaction and the date it was posted. The aggregate can also provide additional information to help the customer recognize the transaction: check number, payee name, and memo. The transaction can have a Standard Industrial Code that a client can use to categorize the transaction.

Each <STMTTRN> contains an <FITID> that the client uses to detect whether the server has previously downloaded the transaction.

Transaction amounts are signed from the perspective of the customer. For example, a credit card payment is positive while a credit card purchase is negative.

Tag	Version	Description
<STMTTRN>		Statement-transaction aggregate
<TRNTYPE>		Transaction type, see section 11.4.3.1 for possible values
<DTPOSTED>		Date transaction was posted to account, ***datetime***

Tag	Version	Description
<DTUSER>		Date user initiated transaction, if known, ***datetime***
<DTAVAIL>		Date funds are available (value date), ***datetime***
<TRNAMT>		Amount of transaction, ***amount***
<FITID>		Transaction ID issued by financial institution. Used to detect duplicate downloads, ***FITID***
<CORRECTFITID>		If present, the FITID of a previously sent transaction that is corrected by this record. This transaction replaces or deletes the transaction that it corrects, based on the value of <CORRECTACTION> below, ***FITID***
<CORRECTACTION>		Actions can be REPLACE or DELETE. REPLACE replaces the transaction referenced by CORRECTFITID; DELETE deletes it.
<SRVRTID>	V1 only	Server assigned transaction ID; used for transactions initiated by client, such as payment or funds transfer. ***SRVRTID***
<SRVRTID2>	V2 only	Server assigned transaction ID; used for transactions initiated by client, such as payment or funds transfer. ***SRVRTID2***
<CHECKNUM>		Check (or other reference) number, ***A-12***
<REFNUM>		Reference number that uniquely identifies the transaction. Can be used in addition to or instead of a <CHECKNUM>, ***A-32***
<SIC>		Standard Industrial Code, ***N-6***
<PAYEEID>	V1 only	Payee identifier if available, ***SRVRTID***
<PAYEEID2>	V2 only	Payee identifier if available, ***SRVRTID2***
Payee options. Choose either <NAME> or <PAYEE>.		
<NAME>		Name of payee or description of transaction, ***A-32***
		Note: Provide NAME or PAYEE, not both
-or-		
<PAYEE>	V1 only	Payee aggregate, see section 12.5.2.1
</PAYEE>		
<PAYEE2>	V2 only	Payee aggregate, see section 12.5.2.1
</PAYEE2>		
Account-to options. Choose either <BANKACCTTO> or <CCACCTTO>.		
<BANKACCTTO>		If this was a transfer to an account and the account information is available, see section 11.3.1
</BANKACCTTO>		
-or-		
<CCACCTTO>		
</CCACCTTO>		
<MEMO>	V1 only	Extra information (not in <NAME>), ***MEMO***
<MEMO2>	V2 only	Extra information (not in <NAME>), ***MEMO2***
Currency options. Choose either <CURRENCY> or <ORIGCURRENCY>.		
<CURRENCY>		Currency, if different from CURDEF
</CURRENCY>		

Tag	Version	Description
-or- **<ORIGCURRENCY>** **</ORIGCURRENCY>**		
</STMTTRN>		

11.4.3.1 Transaction types used in <TRNTYPE>

Type	Description
CREDIT	Generic credit
DEBIT	Generic debit
INT	Interest earned or paid
	Note: depends on signage of amount
DIV	Dividend
FEE	FI fee
SRVCHG	Service charge
DEP	Deposit
ATM	ATM debit or credit
	Note: depends on signage of amount
POS	Point of sale debit or credit
	Note: depends on signage of amount
XFER	Transfer
CHECK	Check
PAYMENT	Electronic payment
CASH	Cash withdrawal
DIRECTDEP	Direct deposit
DIRECTDEBIT	Merchant initiated debit
REPEATPMT	Repeating payment/standing order
OTHER	Other

11.5 Statement Closing Information

Open Financial Exchange provides a way for customers to receive closing statement information that typically appears as part of a paper statement. This information includes opening and closing dates and balances for a statement period, as well as a detailed breakdown of debits, credits, fees, and interest that are usually part of a paper statement. In addition to this information, clients receive a date range for transactions that correspond to the closing statement. Clients might wish to use this date range to retrieve transactions through statement download in order to present the user with an “electronic” statement.

To request statement information, the client is **REQUIRED** to designate an account for the download. The client can also specify a date range to limit the number of closing information aggregates that the server returns. If the client does not specify a date range, the server returns as many closing information aggregates as it can.

Client Sends	Server Responds

Client Sends	*Server Responds*
Account Information	
Date range	
	Cycle-ending information (0 or more)

11.5.1 Statement Closing Download

You can use the <STMTENDRQ> …<STMTENDRS> request and response pair to download statement closing information for checking, savings, money market, and line of credit accounts. Section 11.5.3 describes download for credit card accounts.

11.5.1.1 Request <STMTENDRQ>

The <STMTENDRQ> request must appear within a <STMTENDTRNRQ> transaction wrapper.

Tag	*Description*
<STMTENDRQ>	Closing-statement-request aggregate
<BANKACCTFROM>	Bank-account-from aggregate
</BANKACCTFROM>	
<DTSTART>	Start date for statement closing information, ***datetime***
<DTEND>	End date of statement closing information, ***datetime***
</STMTENDRQ>	

11.5.1.2 Response <STMTENDRS>

The <STMTENDRS> response must appear within a <STMTENDTRNRS> transaction wrapper.

Tag	*Description*
<STMTENDRS>	Closing-statement-response aggregate
<CURDEF>	Default currency used for closing information, ***currsymbol***
<BANKACCTFROM>	Account from aggregate, see section 11.3.1
</BANKACCTFROM>	
<CLOSING>	Statement information (0 or more), see section 11.5.2
</CLOSING>	
</STMTENDRS>	

11.5.1.3 Status Codes

Code	*Meaning*
0	Success
2000	General error (ERROR)
2002	General account error (ERROR)
2003	Account not found (ERROR)
2004	Account closed (ERROR)

Code	Meaning
2005	Account not authorized (ERROR)
2019	Duplicate request (ERROR)

11.5.2 Non-Credit Card Statement <CLOSING>

A checking, savings, or money market account uses the <CLOSING> aggregate to describe statement closing information.

The <FITID> provides a way for the client to distinguish one closing statement from another.

For each <CLOSING> aggregate returned, clients can retrieve corresponding transactions by using <DTPOSTSTART> and <DTPOSTEND> as <DTSTART> and <DTEND> in a <STMTRQ> request.

Tag	Description
<CLOSING>	Non-credit-card-account-types aggregate
<FITID>	Unique identifier for this statement, ***FITID***
<DTOPEN>	Opening statement date, ***date***
<DTCLOSE>	Closing statement date, ***date***
<DTNEXT>	Closing date of next statement, ***date***
<BALOPEN>	Opening statement balance, ***amount***
<BALCLOSE>	Closing statement balance, ***amount***
<BALMIN>	Minimum balance in statement period, ***amount***
<DEPANDCREDIT>	Total of deposits and credits, including interest, ***amount***
<CHKANDDEB>	Total of checks and debits, including fees, ***amount***
<TOTALFEES>	Total of all fees, ***amount***
<TOTALINT>	Total of all interest, ***amount***
<DTPOSTSTART>	Start date of transaction data for this statement, ***date*** A client should be able to use this date in a <STMTRQ> to request transactions that match this statement.
<DTPOSTEND>	End date of transaction data for this statement, ***date*** A client should be able to use this date in a <STMTRQ> to request transactions that match this statement.
<MKTGINFO>	Marketing information (at most 1), ***A-360***
Currency options. Choose either <CURRENCY> or <ORIGCURRENCY>	
<CURRENCY> </CURRENCY> ***-or-*** ***<ORIGCURRENCY> </ORIGCURRENCY>***	Currency, if different from CURDEF
</CLOSING>	

11.5.3 Credit Card Statement Closing Request <CCSTMTENDRQ>

The credit card statement closing request is semantically identical to the bank statement closing request. However, the <CCSTMTENDRQ> aggregate contains the credit card request, not the <STMTENDRQ> aggregate.

The <CCSTMTENDRQ> request must appear within a <CCSTMTENDTRNRQ> transaction wrapper.

Tag	*Description*
<CCSTMTENDRQ>	Credit-card-closing-statement-request aggregate
<CCACCTFROM>	Credit-card-account-from aggregate
</CCACCTFROM>	
<DTSTART>	Start date for statement closing information, ***datetime***
<DTEND>	End date of statement closing information, ***datetime***
</CCSTMTENDRQ>	

11.5.4 Credit Card Statement Closing Response <CCSTMTENDRS>

The credit card statement closing response is semantically identical to the bank statement closing response. However, the <CCSTMTENDRS> aggregate contains the credit card response, not the <STMTENDRS> aggregate.

The <CCSTMTENDRS> response must appear within a <CCSTMTENDTRNRS> transaction wrapper.

Tag	*Description*
<CCSTMTENDRS>	Credit-card-closing-statement-response aggregate
<CURDEF>	Default currency for closing information, ***currsymbol***
<CCACCTFROM>	Account from aggregate, see section 11.3.2
</CCACCTFROM>	
<CCCLOSING>	Statement information (0 or more). See section 11.5.4.2
</CCCLOSING>	
</CCSTMTENDRS>	

11.5.4.1 Status Codes

Code	*Meaning*
0	Success
2000	General error (ERROR)
2002	General account error (ERROR)
2003	Account not found (ERROR)
2004	Account closed (ERROR)
2005	Account not authorized (ERROR)
2019	Duplicate request (ERROR)

11.5.4.2 Credit Card Statement <CCCLOSING>

A credit card account uses the <CCCLOSING> aggregate to describe statement closing information.

The <FITID> provides a way for the client to distinguish one closing statement from another.

For each <CCCLOSING> returned, clients should be able to retrieve corresponding transactions by using <DTPOSTSTART> and <DTPOSTEND> as <DTSTART> and <DTEND> in a <CCSTMTRQ> request.

Tag	*Description*
<CCCLOSING>	Credit-card-statement-information aggregate
<FITID>	Unique identifier for this statement, ***FITID***
<DTOPEN>	Opening statement date, ***date***
<DTCLOSE>	Closing statement date, ***date***
<DTNEXT>	Closing date of next statement, ***date***
<BALOPEN>	Opening statement balance, ***amount***
<BALCLOSE>	Closing statement balance, ***amount***
<DTPMTDUE>	Payment due date, ***date***
<MINPMTDUE>	Minimum amount due, ***amount***
<FINCHG>	Finance charges, ***amount***
<PAYANDCREDIT>	Total of payments and credits, ***amount***
<PURANDADV>	Total of purchases and cash advances, ***amount***
<DEBADJ>	Debit adjustments, ***amount***
<CREDITLIMIT>	Current credit limit, ***amount***
<DTPOSTSTART>	Start date of transaction data for this statement, ***date*** A client should be able to use this date in a <CCSTMTRQ> to request transactions that match this statement.
<DTPOSTEND>	End date of transaction data for this statement, ***date*** A client should be able to use this date in a <CCSTMTRQ> to request transactions that match this statement.
<MKTGINFO>	Marketing information (at most 1), ***A-360***
Currency options. Choose either <CURRENCY> or <ORIGCURRENCY>.	
<CURRENCY> </CURRENCY> ***-or-*** ***<ORIGCURRENCY> </ORIGCURRENCY>***	Currency, if different from CURDEF
</CCCLOSING>	

11.6 Stop Check

Open Financial Exchange supports a request to issue a stop payment for one or more outstanding checks. The stop request can be for a single check or for a range of checks. There must be one request for each check or range of checks the user wants to stop.

When stopping a single check, the client can provide a payee name and optionally an amount instead of a check number to describe the check to stop. Not all servers can support this behavior.

Examples:

Stop check 22	– one request
Stop check to “Acme Lighting”	– one request
Stop checks 200-224	– one request
Stop checks 275-280, 283	– two requests (first stops 275-280, the next stops 283)

Client Sends	*Server Responds*
Account information	
Check number(s) to stop -or- Check description	
	Status for each check

11.6.1 Stop Check Add

Stop Check Add is subject to synchronization.

11.6.1.1 Request <STPCHKRQ>

The <STPCHKRQ> request must appear within a <STPCHKTRNRQ> transaction wrapper.

Tag	*Description*
<STPCHKRQ>	Stop-check-request aggregate
<BANKACCTFROM>	Account-from aggregate, see section 11.3.1
</BANKACCTFROM>	
Check options. Choose either<CHKRANGE> or <CHKDESC>.	
<CHKRANGE>	Check range aggregate, see section 11.6.1.1.1
</CHKRANGE>	
-or-	
<CHKDESC>	Check description aggregate, see section 11.6.1.1.2
</CHKDESC>	
</STPCHKRQ>	

11.6.1.1.1 Check Range <CHKRANGE>

Tag	*Description*
<CHKRANGE>	Check-range aggregate
<CHKNUMSTART>	Start check number to cancel, ***A-12***
<CHKNUMEND>	Ending check number to cancel; omit if only one check is to be stopped, ***A-12***
</CHKRANGE>	

11.6.1.1.2 Check Description <CHKDESC>

A check description must include a payee name or description. It can also include a check number, the date the user wrote the check, and a transaction amount.

Tag	*Description*
<CHKDESC>	Check description aggregate
<NAME>	Payee name or description, ***A-32***
<CHECKNUM>	Check number, ***A-12***
<DTUSER>	Date on check, ***datetime***
<TRNAMT>	Amount, ***amount***
</CHKDESC>	

11.6.1.2 Response <STPCHKRS>

Consistent with all responses, the stop check response contains a global status that describes whether the response could be delivered. If the server provides a response, it returns a <STPCHKNUM> aggregate for each check for which the client requested a stop payment. Status code 10000 should be returned if the stop check request is in process; a subsequent synchronization should obtain an updated response with a final status.

The <STPCHKRS> response must appear within a <STPCHKTRNRS> transaction wrapper.

Tag	*Description*
<STPCHKRS>	Stop-check-response aggregate
<CURDEF>	Default currency for stop check response, ***currsymbol***
<BANKACCTFROM>	Account-from aggregate, see section 11.3.1
</BANKACCTFROM>	
<STPCHKNUM>	Stopped check aggregate (1 or more), see section 11.6.1.2.1
</STPCHKNUM>	
<FEE>	Fee for stop check, ***amount***
<FEEMSG>	Description of fee, ***A-80***
</STPCHKRS>	

11.6.1.2.1 Stopped Check <STPCHKNUM>

This aggregate contains a status code that indicates whether or not a specific check was canceled.

Tag	*Description*
<STPCHKNUM>	Stopped-check-item aggregate
<CHECKNUM>	Check number, ***A-12***
<NAME>	Payee name or description, ***A-32***
<DTUSER>	Date on check, ***datetime***
<TRNAMT>	Amount, ***amount***
<CHKSTATUS>	Status code for individual stop check request 0 = OK 1 = rejected 100 = check not found 101 = check already posted
<CHKERROR>	Further textual explanation, ***A-255***
Currency options. Choose either <CURRENCY> or <ORIGCURRENCY>.	
<CURRENCY> </CURRENCY> ***-or-*** ***<ORIGCURRENCY> </ORIGCURRENCY>***	Currency, if different from CURDEF

11.6.2 Status Codes

Code	*Meaning*
0	Success
2000	General error (ERROR)
2002	General account error (ERROR)
2003	Account not found (ERROR)
2004	Account closed (ERROR)
2005	Account not authorized (ERROR)
2019	Duplicate request (ERROR)
10000	Stop check in process (INFO)
10500	Too many checks to process (ERROR)

11.7 Intrabank Funds Transfer

Open Financial Exchange supports transferring funds between two accounts at the same financial institution. Funds transfers in Open Financial Exchange can be immediate or scheduled. Scheduled transfers can repeat at specified intervals.

Financial institutions can choose to support:

- Immediate transfers
- Immediate and scheduled transfers
- Immediate, scheduled, and recurring transfers

Recurring transfers require support for scheduled transfers.

In general, an OFX server may not choose which transactions to support unless the profile can be used to indicate to the client that a transaction is not supported. However, immediate intrabank funds transfers cannot be modified or canceled, so a server that does not support scheduled transfers may return an error code on any request for cancel or modify. A preferred approach would be to return status code 2016, which means the transfer may not be modified or canceled because it is already committed.

All Intrabank Funds Transfer requests are subject to synchronization.

11.7.1 Intrabank Funds Transfer Addition

The Intrabank Funds Transfer Add request provides a way for a client to set up a single transfer. The request designates source and destination accounts and the amount of the transfer. The client must provide a date if it has scheduled the transfer. Immediate funds transfers cannot be modified or canceled.

Client Sends	*Server Responds*
Source account	
Destination account	
Amount	
Date of transfer (optional)	
	Server ID for the transfer
	Source account
	Destination account
	Amount
	Expected/actual posting date

Intrabank Funds Transfer Add is subject to synchronization.

11.7.1.1 Request <INTRARQ>

The <INTRARQ> request must appear within an <INTRATRNRQ> transaction wrapper.

Tag	*Description*
<INTRARQ>	Intrabank-transfer-request aggregate
<XFERINFO>	Transfer information aggregate, see section 11.3.5
</XFERINFO>	
</INTRARQ>	

11.7.1.2 Response <INTRARS>

A server cannot, in all cases, provide complete confirmation for the transfer. The server can confirm only that it received the transfer instruction; and possibly whether it validated the accounts, amount, and date specified in the transfer. For any transfer where the client does not know the status at the time of the response, a server should confirm that it accepted the instruction and indicate the expected posting date of the transfer. A client can pick up the confirmation at a later date through a synchronization request.

If the request is for an immediate transfer and the server can perform the transfer in real time, the server should indicate whether the transfer succeeded and should return the date of the transfer in <DTPOSTED>. In this case, synchronization is not required.

The <INTRARS> response must appear within an <INTRATRNRS> transaction wrapper.

Tag	*Version*	*Description*
<INTRARS>		Intrabank-transfer-response aggregate
<CURDEF>		Default currency for the intrabank transfer response, ***currsymbol***
<SRVRTID>	V1 only	Server ID for this transfer, ***SRVRTID***
<SRVRTID2>	V2 only	Server ID for this transfer, ***SRVRTID2***
<XFERINFO>		Transfer information aggregate, see section 11.3.5
</XFERINFO>		
Transfer-date options. Choose either <DTXFERPRJ> or <DTPOSTED>		
<DTXFERPRJ>		Projected date of the transfer; response can contain either a <DTXFERPRJ> or a <DTPOSTED> but not both; ***datetime***
-or-		
<DTPOSTED>		Actual date of the transfer, ***datetime***
<RECSRVRTID>	V1 only	If the response is generated by a recurring transfer model, this ID references it, see section 11.10, ***SRVRTID***
<RECSRVRTID2>	V2 only	If the response is generated by a recurring transfer model, this ID references it, see section 11.10, ***SRVRTID2***
<XFERPRCSTS>		Transfer-processing status, see section 11.3.6
</XFERPRCSTS>		
</INTRARS>		

Note: *The server can deliver this response to a client immediately after the request is made (for an immediate or one-time scheduled transfer). The server should also return this response for any transfers that were generated by a model.*

11.7.1.3 Status Codes

Code	*Meaning*
0	Success (INFO)
2000	General error (ERROR)
2002	General account error (ERROR)
2006	Source account not found (ERROR)
2007	Source account closed (ERROR)
2008	Source account not authorized (ERROR)
2009	Destination account not found (ERROR)
2010	Destination account closed (ERROR)

Code	Meaning
2011	Destination account not authorized (ERROR)
2012	Invalid amount (ERROR)
2014	Date too soon (ERROR)
2015	Date too far in future (ERROR)
2019	Duplicate request (ERROR)
10504	Insufficient funds (ERROR)

11.7.2 Intrabank Funds Transfer Modification

The client sends a Transfer Modification request to modify a scheduled transfer. Immediate transfers cannot be modified, so this request should only be used for scheduled transfers. When modifying a transfer, the client must specify all of the tags within the <XFERINFO> aggregate that were specified when the transfer was created, not just the tags that the client wants to modify. <SRVRTID> specifies the transfer the user wants to modify. Not all servers can support the ability to modify some tag values. Clients must not change <BANKACCTFROM> in a funds transfer modification.

Intrabank Funds Transfer Modification is subject to synchronization.

11.7.2.1 Request <INTRAMODRQ>

The <INTRAMODRQ> request must appear within an <INTRATRNRQ> transaction wrapper.

Tag	Version	Description
<INTRAMODRQ>		Modification-request aggregate
<SRVRTID>	V1 only	ID assigned by the server to the transfer being modified, ***SRVRTID***
<SRVRTID2>	V2 only	ID assigned by the server to the transfer being modified, ***SRVRTID2***
<XFERINFO>		Transfer information aggregate, see section 11.3.5
</XFERINFO>		
</INTRAMODRQ>		

11.7.2.2 Response <INTRAMODRS>

This response normally just echoes the values passed by the client. However, if the status of a scheduled transfer changes in any way, clients should expect to receive modification responses when they synchronize with the server. For example, when a server completes a transfer, the status of the transfer goes from *pending* to *posted*. Clients should expect servers to notify them of this status change.

The <INTRAMODRS> response must appear within an <INTRATRNRS> transaction wrapper.

Tag	Version	Description
<INTRAMODRS>		Modification-response aggregate
<SRVRTID>	V1 only	ID assigned by the server to the transfer being modified, ***SRVRTID***
<SRVRTID2>	V2 only	ID assigned by the server to the transfer being modified, ***SRVRTID2***
<XFERINFO>		Transfer information aggregate, see section 11.3.5
</XFERINFO>		
<XFERPRCSTS>		Transfer processing status, see section 11.3.6

Tag	Version	Description
</XFERPRCSTS>		
</INTRAMODRS>		

11.7.2.3 Status Codes

Code	Meaning
0	Success (INFO)
2000	General error (ERROR)
2002	General account error (ERROR)
2006	Source account not found (ERROR)
2007	Source account closed (ERROR)
2008	Source account not authorized (ERROR)
2009	Destination account not found (ERROR)
2010	Destination account closed (ERROR)
2011	Destination account not authorized (ERROR)
2012	Invalid amount (ERROR)
2014	Date too soon (ERROR)
2015	Date too far in future (ERROR)
2016	Already committed (ERROR)
2017	Already canceled (ERROR)
2018	Unknown server ID (ERROR)
2019	Duplicate request (ERROR)
10500	Too many checks to process (ERROR)

11.7.3 Intrabank Funds Transfer Cancellation

The client sends a Transfer Cancellation request to cancel a scheduled transfer, where <SRVRTID> identifies the transfer. Immediate transfers cannot be canceled, so this request should be used only for scheduled transfers.

Intrabank Funds Transfer Cancellation is subject to synchronization.

11.7.3.1 Request <INTRACANRQ>

The <INTRACANRQ> request must appear within an <INTRATRNRQ> transaction wrapper.

Tag	Version	Description
<INTRACANRQ>		Transfer-cancellation-request aggregate
<SRVRTID>	V1 only	ID of the transfer the user wants to cancel. The server must have previously assigned this ID to a transfer. ***SRVRTID***
<SRVRTID2>	V2 only	ID of the transfer the user wants to cancel. The server must have previously assigned this ID to a transfer. ***SRVRTID2***
</INTRACANRQ>		

11.7.3.2 Response <INTRACANRS>

The <INTRACANRS> response must appear within an <INTRATRNRS> transaction wrapper.

Tag	*Version*	*Description*
<INTRACANRS>		Transfer-cancellation-response aggregate
<SRVRTID>	V1 only	ID of the transfer the user wants to cancel. The server must have previously assigned this ID to a transfer. ***SRVRTID***
<SRVRTID2>	V2 only	ID of the transfer the user wants to cancel. The server must have previously assigned this ID to a transfer. ***SRVRTID2***
</INTRACANRQ>		

11.7.3.3 Status Codes

Code	*Meaning*
0	Success
2000	General error (ERROR)
2016	Already committed (ERROR)
2017	Already canceled (ERROR)
2018	Unknown server ID (ERROR)
2019	Duplicate request (ERROR)

11.8 Interbank Funds Transfer

The Interbank Funds Transfer Add request provides a way for a client to set up a single transfer between accounts at different financial institutions. Like intrabank funds transfers, the request designates source and destination accounts and the amount of the transfer. Also, as in intrabank funds transfers, the FI must be able to authenticate the source account. However, interbank funds transfers differ from intrabank funds transfers in the following respects:

- The routing and transit number of the destination account differs from the source account.
- At the discretion of an FI, the destination account can be subject to pre-notification.
- Source and destination accounts must be enabled for the Automated Clearing House (ACH).

Use the ACH system to implement the Interbank Funds Transfer, which is subject to the rules and regulations governing the ACH network.

In all other respects, interbank funds transfers function like intrabank funds transfers. The user can schedule them for a future date or request an immediate transfer. The user can modify or cancel scheduled transfers, but not immediate transfers. Scheduled transfers can recur at regular intervals.

11.8.1 Interbank Funds Transfer – US

In the United States, interbank funds transfers usually use only the <XFERINFO> portion of the request and response.

Client Sends	*Server Responds*
Source account	
Destination account	
Amount	
Date of transfer (optional)	
	Server ID for the transfer
	Source account

Client Sends	Server Responds
	Destination account Amount Expected/actual posting date

Interbank Funds Transfer Add is subject to synchronization.

11.8.2 Interbank Funds Transfer – International Usage

In countries where the funds transfer is the basis of the payments system, the Open Financial Exchange payments messages allow specifying payees by destination account (see Chapter 12, “Payments”).

Interbank Funds Transfer Add is subject to synchronization.

11.8.2.1 Interbank Funds Transfer Request <INTERRQ>

The <INTERRQ> request must appear within an <INTERTRNRQ> transaction wrapper.

Tag	Description
<INTERRQ>	Interbank-transfer-request aggregate
<XFERINFO>	Transfer information aggregate, see section 11.3.5
</XFERINFO>	
</INTERRQ>	

11.8.2.2 Interbank Funds Transfer Response <INTERRS>

The server cannot provide complete confirmation for interbank transfer. It can confirm only that the FI received the transfer instruction and possibly validated the source account, amount, and date specified in the transfer. Since the client does not know the status of the transfer at the time of the response, the server should confirm that it accepted the instruction and indicate the expected posting date of the transfer. The client can pick up the confirmation at a later date through a synchronization request.

The <INTERRS> response must appear within an <INTERTRNRS> transaction wrapper.

Tag	Version	Description
<INTERRS>		Interbank-transfer-response aggregate
<CURDEF>		Currency used in transfer, ***currsymbol***
<SRVRTID>	V1 only	Server ID for this transfer, ***SRVRTID***
<SRVRTID2>	V2 only	Server ID for this transfer, ***SRVRTID2***
<XFERINFO>		Transfer information aggregate, see section 11.3.5
</XFERINFO>		
Transfer-date options. Choose either <DTXFERPRJ> or <DTPOSTED>		
<DTXFERPRJ>		Projected date of the transfer; response can contain either a <DTXFERPRJ> or a <DTPOSTED> but not both; ***datetime***
-or-		
<DTPOSTED>		Actual date of the transfer, ***datetime***
<REFNUM>		Server can generate a reference or check for the transfer, ***A-32***
<RECSRVRTID>	V1 only	If server generates the response by a recurring transfer model, this ID

Tag	Version	Description
		references it. ***SRVRTID***
<RECSRVRTID2>	V2 only	If server generates the response by a recurring transfer model, this ID references it. ***SRVRTID2***
<XFERPRCSTS>		Transfer-processing status, see section 11.3.6
</XFERPRCSTS>		
</INTERRS>		

Note: *A server can deliver this response to a client immediately after the client makes the request (for an immediate or one-time scheduled transfer). In response to a synchronization request by a client, the server should provide a second response containing complete status regarding the transfer. It should also return any transfers that it generates by a model.*

11.8.2.3 Status Codes

Code	Meaning
0	Success
2000	General error (ERROR)
2002	General account error (ERROR)
2006	Source account not found (ERROR)
2007	Source account closed (ERROR)
2008	Source account not authorized (ERROR)
2009	Destination account not found (ERROR)
2010	Destination account closed (ERROR)
2011	Destination account not authorized (ERROR)
2012	Invalid amount (ERROR)
2014	Date too soon (ERROR)
2015	Date too far in future (ERROR)
2019	Duplicate request (ERROR)
10504	Insufficient funds (ERROR)

11.8.3 Interbank Funds Transfer Modification

The client sends a Transfer Modification request to modify a scheduled transfer. Immediate transfers cannot be modified, so this request should only be used for scheduled transfers. When modifying a transfer, the client must specify all of the tags within the <XFERINFO> aggregate that were specified when the transfer was created, not just the tags that the client wants to modify. <SRVRTID> specifies which transfer to modify. Not all servers will support the ability to modify some tag values. Clients must not change <BANKACCTFROM> in a funds transfer modification.

Interbank Funds Transfer Modification is subject to synchronization.

11.8.3.1 Request <INTERMODRQ>

The <INTERMODRQ> request must appear within an <INTERTRNRQ> transaction wrapper.

Tag	Version	Description
<INTERMODRQ>		Modification-request aggregate
<SRVRTID>	V1 only	ID assigned by the server to the transfer being modified, ***SRVRTID***
<SRVRTID2>	V2 only	ID assigned by the server to the transfer being modified, ***SRVRTID2***

Tag	Version	Description
<XFERINFO>		Transfer information aggregate, see section 11.3.5
</XFERINFO>		
</INTERMODRQ>		

11.8.3.2 Response <INTERMODRS>

The <INTERMODRS> response must appear within an <INTERTRNRS> transaction wrapper.

Tag	Version	Description
<INTERMODRS>		Modification-response aggregate
<SRVRTID>	V1 only	ID assigned by the server to the transfer being modified, ***SRVRTID***
<SRVRTID2>	V2 only	ID assigned by the server to the transfer being modified, ***SRVRTID2***
<XFERINFO>		Transfer information aggregate; server returns if client provided an <XFERINFO> in the request, see section 11.3.5
</XFERINFO>		
<XFERPRCSTS**>**		Processing status for transfer, see section 11.3.6
</XFERPRCSTS**>**		
</INTERMODRS>		

11.8.3.3 Status Codes

Code	Meaning
0	Success
2000	General error (ERROR)
2002	General account error (ERROR)
2006	Source account not found (ERROR)
2007	Source account closed (ERROR)
2008	Source account not authorized (ERROR)
2009	Destination account not found (ERROR)
2010	Destination account closed (ERROR)
2011	Destination account not authorized (ERROR)
2012	Invalid amount (ERROR)
2014	Date too soon (ERROR)
2015	Date too far in future (ERROR)
2016	Already committed (ERROR)
2017	Already canceled (ERROR)
2018	Unknown server ID (ERROR)
2019	Duplicate request (ERROR)
10504	Insufficient funds (ERROR)
10505	Cannot modify element (ERROR)

11.8.4 Interbank Funds Transfer Cancellation

The client sends a Transfer Cancellation request to cancel a scheduled interbank transfer, where <SRVRTID> identifies the transfer. Immediate transfers cannot be canceled, so this request should only be used for scheduled transfers.

Interbank Funds Transfer Cancellation is subject to synchronization.

11.8.4.1 Request <INTERCANRQ>

The <INTERCANRQ> request must appear within an <INTERTRNRQ> transaction wrapper.

Tag	*Version*	*Description*
<INTERCANRQ>		Transfer-cancellation-request aggregate
<SRVRTID>	V1 only	ID of the transfer to cancel. The server must have previously assigned this ID to a transfer. ***SRVRTID***
<SRVRTID2>	V2 only	ID of the transfer to cancel. The server must have previously assigned this ID to a transfer. ***SRVRTID2***
</INTERCANRQ>		

11.8.4.2 Response <INTERCANRS>

The <INTERCANRS> response must appear within an <INTERTRNRS> transaction wrapper.

Tag	*Version*	*Description*
<INTERCANRS>		Transfer-cancellation-response aggregate
<SRVRTID>	V1 only	ID of the transfer to cancel. The server must have previously assigned this ID to a transfer. ***SRVRTID***
<SRVRTID2>	V2 only	ID of the transfer to cancel. The server must have previously assigned this ID to a transfer. ***SRVRTID2***
</INTERCANRS>		

11.8.4.3 Status Codes

Code	*Meaning*
0	Success
2000	General error (ERROR)
2016	Already committed (ERROR)
2017	Already canceled (ERROR)
2018	Unknown server ID (ERROR)
2019	Duplicate request (ERROR)

11.8.5 Multiple Interbank Funds Transfer

Some countries have a special interbank funds transfer transaction. Such a transaction combines multiple interbank transfer instructions into a single unit of work. Open Financial Exchange has a special transaction wrapper, the <MULTIINTERTRNRQ>, that can contain 1 or more interbank transfer aggregates. Only Message set version 2 and above support <MULTIINTERTRNRQ>, and even then only if the FI profile says it's supported.

11.8.5.1 Multiple Interbank Funds Transfer Request <MULTIINTERTRNRQ>

The <MULTIINTERTRNRQ> wrapper contains 1 or more <INTERRQ> aggregates. The interbank transfers are a single unit of work that share a common <TRNUID>, <CLTCOOKIE> and <TAN>. In other words if one of the requests within the transaction wrapper fails then all the request fail.

Open Financial Exchange does not support interbank transfer modify and interbank transfer cancellation requests in the <MULTIINTERTRNRQ>.

Tag	*Description*
<MULTIINTERTRNRQ>	Multiple interbank-transfer-transaction wrapper.
<TRNUID>	
<CLTCOOKIE>	
<TAN>	
<INTERRQ>	One or more interbank transfer request aggregates, see section 11.8.2.1.
</INTERRQ>	
</MULTIINTERTRNRQ>	

11.8.5.2 Multiple Interbank Funds Transfer Response <MULTIINTERTRNRS>

The <MULTIINTERTRNRS> wrapper contains 1 or more <INTERRS>aggregates. The interbank transfers share a common <TRNUID>, <CLTCOOKIE> and <STATUS>.

Tag	*Description*
<MULTIINTERTRNRS>	Multiple interbank-transfer-transaction wrapper.
<TRNUID>	
<STATUS>	
</STATUS>	
<CLTCOOKIE>	
<INTERRS>	One or more interbank transfer response aggregates, see section 11.8.2.2.
</INTERRS>	
</MULTIINTERTRNRS>	

11.8.5.3 Status Codes

Code	*Meaning*
0	Success
2000	General error (ERROR)
2002	General account error (ERROR)

Code	Meaning
2006	Source account not found (ERROR)
2007	Source account closed (ERROR)
2008	Source account not authorized (ERROR)
2009	Destination account not found (ERROR)
2010	Destination account closed (ERROR)
2011	Destination account not authorized (ERROR)
2012	Invalid amount (ERROR)
2014	Date too soon (ERROR)
2015	Date too far in future (ERROR)
2019	Duplicate request (ERROR)
10504	Insufficient funds (ERROR)

11.9 Wire Funds Transfer

Open Financial Exchange enables clients to set up wire funds transfers. Wire funds transfers are similar to other types of funds transfers. Clients designate a source account that the FI can authenticate and a destination account at the same or a different institution. Clients also designate an amount and an optional date.

The FI must know the originator of the transfer. The beneficiary of the transfer might be an established customer at the same institution.

Open Financial Exchange implements wire funds transfers using the FedWire system, and is subject to its rules and regulations.

In almost all respects, wire funds transfers work like interbank funds transfers. A user can schedule and cancel them. Unlike interbank funds transfers, a user cannot modify Wire funds transfers once they have been set up. A user cannot set up wire funds transfers to recur at regular intervals.

Client Sends	Server Responds
Source account	
Originator	
Receiver	
Amount	
Date of transfer (optional)	
	Server ID for the transfer
	Originator
	Receiver
	Amount
	Expected/actual posting date

11.9.1 Wire Funds Transfer Addition

Wire Funds Transfer Add is subject to synchronization.

11.9.1.1 Request <WIRERQ>

The client prepares a <BANKACCTFROM> aggregate to describe the source account. The <WIREBENEFICIARY> aggregate specifies the destination account. The <WIREDESTBANK> aggregate describes the beneficiary's bank.

The <WIRERQ> request must appear within a <WIRETRNRQ> transaction wrapper.

Tag	*Description*
<WIRERQ>	Wire-transfer-request aggregate
<BANKACCTFROM>	Source of funds
</BANKACCTFROM>	
<WIREBENEFICIARY>	Wire transfer beneficiary, see section 11.9.1.1.1
</WIREBENEFICIARY>	
<WIREDESTBANK>	Beneficiary's bank
<EXTBANKDESC>	Extended bank description, see section 11.9.1.1.2
</EXTBANKDESC>	
</WIREDESTBANK>	
<TRNAMT>	Transfer amount, ***amount***
<DTDUE>	Date to occur, ***datetime***
<PAYINSTRUCT>	Payment instructions, ***A-255***
</WIRERQ>	

11.9.1.1.1 Wire Beneficiary Aggregate <WIREBENEFICIARY>

The wire beneficiary aggregate describes the receiver of a wire transfer.

Tag	*Version*	*Description*
<WIREBENEFICIARY>		Wire-beneficiary aggregate
<NAME>		Name of beneficiary, ***A-32***
<BANKACCTTO>		Bank details for beneficiary
</BANKACCTTO>		
<MEMO>	V1 only	Information for the beneficiary, ***memo***
<MEMO2>	V2 only	Information for the beneficiary, ***memo2***
</WIREBENEFICIARY>		

11.9.1.1.2 Extended Bank Description aggregate <EXTBANKDESC>

Tag	*Description*
<EXTBANKDESC>	Extended-bank-description aggregate
<NAME>	Abbreviated name of bank, ***A-32***
<BANKID>	Routing: ABA number or S.W.I.F.T. number, ***A-9***

Tag	Description
<ADDR1>	Bank's address line 1, ***A-32***
<ADDR2>	Bank's address line 2, ***A-32***
<ADDR3>	Bank's address line 3, ***A-32***
<CITY>	Bank's city, ***A-32***
<STATE>	Bank's state or province, ***A-5***
<POSTALCODE>	Bank's zip code, ***A-11***
<COUNTRY>	Bank's country; 3-letter country code from ISO/DIS-3166, ***A-3***
<PHONE>	Bank's phone number, ***A-32***
</EXTBANKDESC>	

11.9.1.2 Response <WIRERS>

The server cannot provide complete confirmation for the transfer. It can confirm only that the server received the transfer instruction and possibly that it validated the source account, amount, and date specified in the transfer. For any transfer where the client does not know the status at the time of the response, the server should confirm that it accepted the instruction and indicate the expected posting date of the transfer. The client can pick up the confirmation at a later date through a synchronization request.

The server can indicate the fee assessed for the transfer by using the <FEE> element in the response. The server can also include a confirmation message in the response.

The <WIRERS> response must appear within a <WIRETRNRS> transaction wrapper.

Tag	Version	Description
<WIRERS>		Wire-transfer-response aggregate
<CURDEF>		Currency used in transfer, ***currsymbol***
<SRVRTID>	V1 only	Server ID for this transfer, ***SRVRTID***
<SRVRTID2>	V2 only	Server ID for this transfer, ***SRVRTID2***
<BANKACCTFROM>		Source of funds
</BANKACCTFROM>		
<WIREBENEFICIARY>		Wire transfer beneficiary, see section 11.9.1.1.1
</WIREBENEFICIARY>		
<WIREDESTBANK>		Beneficiary's bank
<EXTBANKDESC>		Extended bank description, see section 11.9.1.1.2
</EXTBANKDESC>		
</WIREDESTBANK>		
<TRNAMT>		Transfer amount, ***amount***
<DTDUE>		Date to occur, echoed if provided in request, ***datetime***
<PAYINSTRUCT>		Payment instructions, echoed if provided in request, ***A-255***
Transfer-date options. Choose either <DTXFERPRJ> or <DTPOSTED>		
<DTXFERPRJ>		Projected date of the transfer; response can contain either a <DTXFERPRJ> or a <DTPOSTED> but not both; ***datetime***
-or-		
<DTPOSTED>		Actual date of the transfer, ***datetime***
<FEE>		Fee assessed for the transfer, ***amount***

Tag	Version	Description
<CONFMSG>		Confirmation message, ***A-255***
</WIRERS>		

11.9.1.3 Status Codes

Code	Meaning
0	Success
2000	General error (ERROR)
2002	General account error (ERROR)
2006	Source account not found (ERROR)
2007	Source account closed (ERROR)
2008	Source account not authorized (ERROR)
2012	Invalid amount (ERROR)
2014	Date too soon (ERROR)
2015	Date too far in future (ERROR)
2019	Duplicate request (ERROR)
10504	Insufficient funds (ERROR)
10516	Wire beneficiary invalid (ERROR)

11.9.2 Wire Funds Transfer Cancellation

The client sends a Wire Funds Transfer Cancellation Request to cancel a scheduled transfer, where <SRVRTID> identifies the transfer.

Wire Funds Transfer Cancellation is subject to synchronization.

11.9.2.1 Request <WIRECANRQ>

The <WIRECANRQ> request must appear within a <WIRETRNRQ> transaction wrapper.

Tag	Version	Description
<WIRECANRQ>		Wire-transfer-cancellation-request aggregate
<SRVRTID>	V1 only	ID of the transfer to cancel; server must have previously assigned this ID to a transfer, ***SRVRTID***
<SRVRTID2>	V2 only	ID of the transfer to cancel; server must have previously assigned this ID to a transfer, ***SRVRTID2***
</WIRECANRQ>		

11.9.2.2 Response <WIRECANRS>

The <WIRECANRS> response must appear within a <WIRETRNRS> transaction wrapper.

Tag	Version	Description
<WIRECANRS>		Wire-transfer-cancellation-response aggregate
<SRVRTID>	V1 only	ID of the transfer to cancel; server must have previously assigned this ID to a transfer, ***SRVRTID***

Tag	Version	Description
<SRVRTID2> **</WIRECANRS>**	V2 only	ID of the transfer to cancel; server must have previously assigned this ID to a transfer, ***SRVRTID2***

11.9.2.3 Status Codes

Code	Meaning
0	Success
2000	General error (ERROR)
2016	Already committed (ERROR)
2017	Already canceled (ERROR)
2019	Duplicate request (ERROR)

11.10 Recurring Funds Transfer

Open Financial Exchange uses a Recurring Funds Transfer Add request to set up a recurring transfer model. The transfer model generates transfers according to its schedule. Transfers created by a model and retrieved by a customer can be modified or canceled without impacting the model.

A user can create recurring funds transfer models to generate two types of scheduled transfers: interbank and intrabank. You cannot set up recurring wire funds transfers.

For more information on recurring transactions, see Chapter 10, “Recurring Transactions.”

11.10.1 Recurring Intrabank Funds Transfer Addition

A Recurring Intrabank Funds Transfer Add request sets up an intrabank funds transfer that repeats at a specified interval for a specified period of time.

Model-created transfers are retrieved by means of a synchronization request.

Client Sends	Server Responds
Source account	
Destination account	
Amount	
Date of first transfer	
Frequency	
Duration	
	Server ID for the model
	Source account
	Destination account
	Amount
	Date of first transfer
	Frequency
	Duration

Recurring Intrabank Funds Transfer Add is subject to synchronization.

11.10.1.1 Request <RECINTRARQ>

The <RECINTRARQ> request must appear within a <RECINTRATRNRQ> transaction wrapper.

Tag	*Version*	*Description*
<RECINTRARQ>		Recurring-transfer-request aggregate
<RECURRINST>		Recurring-instructions aggregate, see section
</RECURRINST>		
<INTRARQ>	V1 only	Intrabank-transfer-request aggregate, see section 11.7.1.1
</INTRARQ>		
<XFERINFO>	V2 only	Transfer info aggregate, see section 11.3.5.
		Version 2 uses <XFERINFO> instead of <INTRARQ>
</XFERINFO>		
</RECINTRARQ>		

11.10.1.2 Response <RECINTRARS>

The <RECINTRARS> response must appear within a <RECINTRATRNRS> transaction wrapper.

For version 1 of the message set, the <SRVRTID> included in the <INTRARS> should be set to the same value as the <RECSRVRTID>.

> ***Note:*** *this is the response to the recurring model only. Servers must still generate an INTRARS for each instance of the recurring transfer.*

Tag	*Version*	*Description*
<RECINTRARS>		Recurring-transfer-response aggregate
<RECSRVRTID>	V1 only	Server-assigned ID for this model, ***SRVRTID***
<RECSRVRTID2>	V2 only	Server-assigned ID for this model, ***SRVRTID2***
<RECURRINST>		Recurring-instructions aggregate
</RECURRINST>		
<INTRARS>	V1 only	Intrabank-transfer-response aggregate, see section 11.7.1.2
</INTRARS>		
<XFERINFO>	V2 only	Transfer Info aggregate, see section 11.3.5
		Version 2 uses <XFERINFO> instead of <INTRARS>
</XFERINFO>		
</RECINTRARS>		

11.10.1.3 Status Codes

Code	*Meaning*
0	Success

Code	Meaning
2000	General error (ERROR)
2002	General account error (ERROR)
2006	Source account not found (ERROR)
2007	Source account closed (ERROR)
2008	Source account not authorized (ERROR)
2009	Destination account not found (ERROR)
2010	Destination account closed (ERROR)
2011	Destination account not authorized (ERROR)
2014	Date too soon (ERROR)
2015	Date too far in future (ERROR)
2019	Duplicate request (ERROR)
10508	Invalid frequency (ERROR)

11.10.2 Recurring Intrabank Funds Transfer Modification

The client sends a Recurring Intrabank Funds Transfer Modification request to modify a recurring intrabank transfer model.

Recurring Intrabank Funds Transfer Modification is subject to synchronization.

Clients must not change <BANKACCTFROM> in a recurring funds transfer modification.

11.10.2.1 Request <RECINTRAMODRQ>

<RECSRVRTID> identifies the model. The client can indicate whether the changes should apply to pending transfers.

The <RECINTRAMODRQ> request must appear within a <RECINTRATRNRQ> transaction wrapper.

Tag	Version	Description
<RECINTRAMODRQ>		Recurring-modification-request aggregate
<RECSRVRTID>	V1 only	ID assigned by the server to the model being modified, ***SRVRTID***
<RECSRVRTID2>	V2 only	ID assigned by the server to the model being modified, ***SRVRTID2***
<RECURRINST>		Recurring-instructions aggregate
</RECURRINST>		
<INTRARQ>	V1 only	Intrabank-transfer-request aggregate, see section 0
</INTRARQ>		
<XFERINFO>	V2 only	Transfer info aggregate, see section 0. Version 2 uses <XFERINFO> instead of <INTRARQ>
</XFERINFO>		
<MODPENDING>		Modify pending flag, ***Boolean*** If the client sets this flag, the server must modify pending and future transfers.
</RECINTRAMODRQ>		

11.10.2.2 Response <RECINTRAMODRS>

The <RECINTRAMODRS> response must appear within a <RECINTRATRNRS> transaction wrapper.

Tag	*Version*	*Description*
<RECINTRAMODRS>		Recurring-transfer-modification-request aggregate
<RECSRVRTID>	V1 only	ID assigned by the server to the model being modified, ***SRVRTID***
<RECSRVRTID2>	V2 only	ID assigned by the server to the model being modified, ***SRVRTID2***
<RECURRINST>		Recurring-instructions aggregate
</RECURRINST>		
<INTRARS>	V1 only	Intrabank transfer response aggregate, see section 11.7.1.2
</INTRARS>		
<XFERINFO>	V2 only	Transfer Info aggregate, see section 0 Version 2 uses <XFERINFO> instead of <INTRARS>
</XFERINFO>		
<MODPENDING>		Y if client requested that the server modify pending and future transfers. N if the client did not request that the server modify pending and future transfers. ***Boolean***
</RECINTRAMODRS>		

11.10.2.3 Status Codes

Code	*Meaning*
0	Success (INFO)
2000	General error (ERROR)
2002	General account error (ERROR)
2006	Source account not found (ERROR)
2007	Source account closed (ERROR)
2008	Source account not authorized (ERROR)
2009	Destination account not found (ERROR)
2010	Destination account closed (ERROR)
2011	Destination account not authorized (ERROR)
2012	Invalid amount (ERROR)
2014	Date too soon (ERROR)
2015	Date too far in future (ERROR)
2016	Already committed (ERROR)
2017	Already canceled (ERROR)
2019	Duplicate request (ERROR)
10500	Too many checks to process (ERROR)
10505	Cannot modify element (ERROR)
10508	Invalid frequency (ERROR)
10518	Unknown model ID (ERROR)

11.10.3 Recurring Intrabank Funds Transfer Cancellation

The client sends a Recurring Intrabank Funds Transfer Cancellation request to cancel a recurring intrabank transfer model.

Recurring Intrabank Funds Transfer Cancellation is subject to synchronization.

11.10.3.1 Request <RECINTRACANRQ>

<RECSRVRTID> identifies the model the user wants to cancel. The client can indicate whether the cancel should apply to pending transfers.

The <RECINTRACANRQ> request must appear within a <RECINTRATRNRQ> transaction wrapper.

Tag	*Version*	*Description*
<RECINTRACANRQ>		Recurring-transfer-cancellation-request aggregate
<RECSRVRTID>	V1 only	ID assigned by the server to the model being canceled, ***SRVRTID***
<RECSRVRTID2>	V2 only	ID assigned by the server to the model being canceled, ***SRVRTID2***
<CANPENDING>		Cancel pending flag, ***Boolean***
		If the client sets this flag, the server must cancel pending and future transfers.
</RECINTRACANRQ>		

11.10.3.2 Response <RECINTRACANRS>

The <RECINTRACANRS> response must appear within a <RECINTRATRNRS> transaction wrapper.

Tag	*Version*	*Description*
<RECINTRACANRS>		Recurring-transfer-cancellation-response aggregate
<RECSRVRTID>	V1 only	ID assigned by the server to the model being canceled, ***SRVRTID***
<RECSRVRTID2>	V2 only	ID assigned by the server to the model being canceled, ***SRVRTID2***
<CANPENDING>		Cancel pending flag, ***Boolean***
		Y if the client requested that the server cancel pending and future transfers. N if the client did not request that the server cancel pending and future transfers.
</RECINTRACANRS>		Ending tag for recurring transfer cancellation response

11.10.3.3 Status Codes

Code	*Meaning*
0	Success
2000	General error (ERROR)
2019	Duplicate request (ERROR)
10509	Model already canceled (ERROR)
10518	Unknown model ID (ERROR)

11.10.4 Recurring Interbank Funds Transfer Addition

A Recurring Interbank Funds Transfer Add request sets up an interbank funds transfer that repeats at a specified interval for a specified period of time.

The client retrieves model-created transfers by means of a synchronization request.

Client Sends	*Server Responds*
Source account	
Destination account	
Amount	
Date of first transfer	
Frequency	
Duration	
	Server ID for the model
	Source account
	Destination account
	Amount
	Date of first transfer
	Frequency
	Duration

Recurring Interbank Funds Transfer Add is subject to synchronization

11.10.4.1 Request <RECINTERRQ>

The <RECINTERRQ> request must appear within a <RECINTERTRNRQ> transaction wrapper.

Tag	*Version*	*Description*
<RECINTERRQ>		Recurring-transfer-request aggregate
<RECURRINST>		Recurring-instructions aggregate
</RECURRINST>		
<INTERRQ>	V1 only	Interbank-transfer-request aggregate, see section 0
</INTERRQ>		
<XFERINFO>	V2 only	Transfer info aggregate, see section 0.
		Version 2 uses <XFERINFO> instead of <INTERRQ>
</XFERINFO>		
</RECINTERRQ>		

11.10.4.2 Response <RECINTERRS>

The <RECINTERRS> response must appear within a <RECINTERTRNRS> transaction wrapper.

Tag	*Version*	*Description*
<RECINTERRS>		Recurring-transfer-response aggregate

Tag	Version	Description
<RECSRVRTID>	V1 only	Server-assigned ID for this model, ***SRVRTID***
<RECSRVRTID2>	V2 only	Server-assigned ID for this model, ***SRVRTID2***
<RECURRINST>		Recurring-instructions aggregate, see section 10.2
</RECURRINST>		
<INTERRS>	V1 only	Interbank funds transfer response, see section 11.8.2.2
</INTERRS>		
<XFERINFO>	V2 only	Transfer Info aggregate, see section 0
		Version 2 uses <XFERINFO> instead of <INTERRS>
</XFERINFO>		
</RECINTERRS>		

11.10.4.3 Status Codes

Code	Meaning
0	Success
2000	General error (ERROR)
2002	General account error (ERROR)
2006	Source account not found (ERROR)
2007	Source account closed (ERROR)
2008	Source account not authorized (ERROR)
2009	Destination account not found (ERROR)
2010	Destination account closed (ERROR)
2011	Destination account not authorized (ERROR)
2012	Invalid amount (ERROR)
2014	Date too soon (ERROR)
2015	Date too far in future (ERROR)
2019	Duplicate request (ERROR)
10504	Insufficient funds (ERROR)
10508	Invalid frequency (ERROR)

11.10.5 Recurring Interbank Funds Transfer Modification

The client sends a Recurring Interbank Funds Transfer Modification request to modify a recurring interbank transfer model.

Recurring Interbank Funds Transfer Modification is subject to synchronization.

Clients must not change <BANKACCTFROM> in a recurring funds transfer modification.

11.10.5.1 Request <RECINTERMODRQ>

<RECSRVRTID> identifies the model. The client can indicate whether the changes should apply to pending transfers.

The <RECINTERMODRQ> request must appear within a <RECINTERTRNRQ> transaction wrapper.

Tag	*Version*	*Description*
<RECINTERMODRQ>		Recurring-modification-request aggregate
<RECSRVRTID>	V1 only	ID assigned by the server to the model being modified, ***SRVRTID***
<RECSRVRTID2>	V2 only	ID assigned by the server to the model being modified, ***SRVRTID2***
<RECURRINST>		Recurring-instructions aggregate
</RECURRINST>		
<INTERRQ>	V1 only	Interbank-funds-transfer-request aggregate, see section 0
</INTERRQ>		
<XFERINFO>	V2 only	Transfer info aggregate, see section 0. Version 2 uses <XFERINFO> instead of <INTERRQ>
</XFERINFO>		
<MODPENDING>		Modify pending flag If the client sets this flag, the server must modify pending and future transfers. ***Boolean***
</RECINTERMODRQ>		

11.10.5.2 Request <RECINTERMODRS>

The <RECINTERMODRS> response must appear within a <RECINTERTRNRS> transaction wrapper.

Tag	*Version*	*Description*
<RECINTERMODRS>		Recurring-transfer-modification-response aggregate
<RECSRVRTID>	V1 only	ID assigned by the server to the model being modified, ***SRVRTID***
<RECSRVRTID2>	V2 only	ID assigned by the server to the model being modified, ***SRVRTID2***
<RECURRINST>		Recurring-instructions aggregate
</RECURRINST>		
<INTERRS>	V1 only	Interbank-funds-transfer-response, see section 11.8.2.2
</INTERRS>		
<XFERINFO>	V2 only	Transfer Info aggregate, see section 0 Version 2 uses <XFERINFO> instead of <INTERRS>
</XFERINFO>		
<MODPENDING>		Modify pending flag, ***Boolean*** Y if the client requested that the server modify pending and future transfers. N if the client did not request that the server modify pending and future transfers.

Tag	Version	Description
</RECINTERMODRS>		

11.10.5.3 Status Codes

Code	Meaning
0	Success
2000	General error (ERROR)
2002	General account error (ERROR)
2006	Source account not found (ERROR)
2007	Source account closed (ERROR)
2008	Source account not authorized (ERROR)
2009	Destination account not found (ERROR)
2010	Destination account closed (ERROR)
2011	Destination account not authorized (ERROR)
2012	Invalid amount (ERROR)
2014	Date too soon (ERROR)
2015	Date too far in future (ERROR)
2016	Already committed (ERROR)
2017	Already canceled (ERROR)
2019	Duplicate request (ERROR)
10504	Insufficient funds (ERROR)
10505	Cannot modify element (ERROR)
10508	Invalid frequency (ERROR)
10510	Invalid payee ID (ERROR)
10518	Unknown model ID (ERROR)

11.10.6 Recurring Interbank Funds Transfer Cancellation

The client sends a Recurring Transfer Cancellation request to cancel a recurring transfer model.

Recurring Transfer Cancellation is subject to synchronization.

11.10.6.1 Request <RECINTERCANRQ>

<RECSRVRTID> identifies the model the client wants to cancel. The client can indicate whether the cancel should apply to pending transfers.

The <RECINTERCANRQ> request must appear within a <RECINTERTRNRQ> transaction wrapper.

Tag	Version	Description
<RECINTERCANRQ>		Recurring-transfer-cancellation-request aggregate
<RECSRVRTID>	V1 only	ID assigned by the server to the model being canceled, ***SRVRTID***
<RECSRVRTID2>	V2 only	ID assigned by the server to the model being canceled, ***SRVRTID2***

Tag	Version	Description
<CANPENDING>		Cancel pending flag, ***Boolean*** If the client sets this flag, the server must cancel pending and future transfers.
</RECINTERCANRQ>		

11.10.6.2 Response <RECINTERCANRS>

The <RECINTERCANRS> response must appear within a <RECINTERTRNRS> transaction wrapper.

Tag	Version	Description
<RECINTERCANRS>		Recurring-transfer-cancellation-response aggregate
<RECSRVRTID>	V1 only	ID assigned by the server to the model being canceled, ***SRVRTID***
<RECSRVRTID2>	V2 only	ID assigned by the server to the model being canceled, ***SRVRTID2***
<CANPENDING>		Cancel pending flag, ***Boolean*** Y if the client requested that the server cancel pending and future transfers. N if the client did not request that the server cancel pending and future transfers
</RECINTERCANRS>		

11.10.6.3 Status Codes

Code	Meaning
0	Success
2000	General error (ERROR)
2019	Duplicate request (ERROR)
10509	Model already canceled (ERROR)
10518	Unknown model ID (ERROR)

11.11 E-Mail and Customer Notification

Open Financial Exchange enables customers to contact their FIs when they have questions regarding their accounts. FIs can also notify their customers of significant events that have occurred regarding their accounts. For example, notification can occur if a customer writes a check that does not clear due to insufficient funds. The server prepares the notification and the client picks it up the next time it synchronizes with the server.

11.11.1 Banking E-Mail

Open Financial Exchange currently defines one banking e-mail message that clients can send to an FI. With this message, the user can prepare a message to the FI regarding one of his accounts. The server acknowledges receipt of the message. The FI prepares the response that the client picks up when it synchronizes with the server.

Client Sends	Server Responds
Addressed message Bank account information	

Client Sends	Server Responds
	Acknowledgment
.	
.	
.	
Synchronization request	
	Response to customer

11.11.1.1 Request <BANKMAILRQ>

The client must identify to which bank account the customer query is related.

The <BANKMAILRQ> request must appear within a <BANKMAILTRNRQ> transaction wrapper.

Tag	Description
<BANKMAILRQ>	Bank-e-mail-request aggregate
Account-from options. Choose either <BANKACCTFROM> or <CCACCTFROM>.	
<BANKACCTFROM>	Account-from aggregate, see section 11.3.1
</BANKACCTFROM>	
-or-	
<CCACCTFROM>	Credit-card-account-from aggregate, see section 11.3.2
</CCACCTFROM>	
<MAIL>	To, from, message information, see Chapter 9, "Customer to FI Communication"
</MAIL>	
</**BANKMAILRQ**>	

11.11.1.2 Response <BANKMAILRS>

The <BANKMAILRS> response must appear within a <BANKMAILTRNRS> transaction wrapper.

Tag	Description
<BANKMAILRS>	Bank-e-mail-response aggregate
Account-from options. Choose either <BANKACCTFROM> or <CCACCTFROM>.	
<BANKACCTFROM>	Account-from aggregate, see section 11.3.1
</BANKACCTFROM>	
-or-	
<CCACCTFROM>	Credit-card-account-from aggregate, see section 11.3.2
</CCACCTFROM>	
<MAIL>	To, from, message information, see Chapter 9, "Customer to FI Communication
</MAIL>	
</**BANKMAILRS**>	

11.11.1.3 Status Codes

Code	Meaning
0	Success (INFO)
2000	General error (ERROR)
2002	General account error (ERROR)
2003	Account not found (ERROR)
2004	Account closed (ERROR)
2005	Account not authorized (ERROR)
2019	Duplicate request (ERROR)
16500	HTML not allowed (ERROR)
16501	Unknown mail To: (ERROR)

11.11.2 Notifications

Open Financial Exchange currently defines two banking notifications that an FI can support:

- Returned check
- Returned deposit

You can implement banking notifications through e-mail and synchronization. The client provides a <TOKEN> representing its current state with regard to banking notification. (See section 3.2.4.) The server can respond by returning a new token and one or more notification e-mail responses.

Client Sends	Server Responds
Synchronization request with current token	
	New token
	Bank e-mail
	Mail for returned check
	Mail for returned deposit

11.11.3 Returned Check and Deposit Notification

11.11.3.1 Response <CHKMAILRS>

The server returns this response (when a check has been returned), if it receives a banking e-mail synchronization message.

The <CHKMAILRS> response must appear within a <BANKMAILTRNRS> transaction wrapper.

Tag	Description
<CHKMAILRS>	Notification-message-response aggregate
<BANKACCTFROM>	Account-from aggregate, see section 11.3.1
</BANKACCTFROM>	
<MAIL>	To, from, message information, see Chapter 9, “Customer to FI Communication
</MAIL>	
<CHECKNUM>	Check number, ***A-12***

Tag	Description
<TRNAMT>	Amount of check, ***amount***
<DTUSER>	Customer date on check, ***date***
<FEE>	Fee assessed for NSF, ***amount***
</CHKMAILRS>	

11.11.3.2 Response <DEPMAILRS>

The server returns this response (when a deposit has been returned), if it receives a banking e-mail synchronization message.

The <DEPMAILRS> response must appear within a <BANKMAILTRNRS> transaction wrapper.

Tag	Description
<DEPMAILRS>	Notification-message-response aggregate
<BANKACCTFROM>	Account-from aggregate, see section 11.3.1
</BANKACCTFROM>	
<MAIL>	To, from, message information, see Chapter 9, "Customer to FI Communication
</MAIL>	
<TRNAMT>	Amount of deposit, ***amount***
<DTUSER>	Customer date of deposit, ***date***
<FEE>	Fee assessed for NSF, ***amount***
</DEPMAILRS>	

11.12 Data Synchronization for Banking

Banking customers must be able to obtain the current status of transactions previously sent to the server for processing. For example, once a client schedules a transfer and the transfer date has passed, the customer might wish to verify that the server made the transfer as directed. Also, Open Financial Exchange allows for interactions with the server through multiple clients. This means, for example, that the customer can perform some transactions from a home PC and others from an office computer, with each session seamlessly incorporating the activities performed on the other.

To accomplish these actions, the client uses a synchronization scheme to ensure that it has an accurate copy of the server data that is relevant to the client application.

Banking requires synchronization in the following areas: Stop Check, IntraBank Transfers, InterBank Transfers, Wire Transfers, and Banking Notifications.

11.12.1 Data Synchronization for Stop Check

11.12.1.1 Request <STPCHKSYNCRQ>

Tag	Version	Description
<STPCHKSYNCRQ>		Synchronization-request aggregate
Client synchronization option; <TOKEN>, <TOKEN2>, <TOKENONLY>, or <REFRESH>		

Tag	Version	Description
<TOKEN>	V1 only	Previous value of <TOKEN> received for this type of synchronization request from server; 0 for first-time requests; ***token***
<TOKEN2>	V2 only	Previous value of <TOKEN2> received for this type of synchronization request from server; 0 for first-time requests; ***token2***
<TOKENONLY>		Request for just the current <TOKEN> without the history, ***Boolean***
<REFRESH>		Request for refresh of current state, ***Boolean***
<REJECTIFMISSING>		If Y, do not process requests if client <TOKEN> is out of date, ***Boolean***
<BANKACCTFROM>		Bank account of interest; token must be interpreted in terms of this account
</BANKACCTFROM>		
<STPCHKTRNRQ>		Stop-check transactions (0 or more)
</STPCHKTRNRQ>		
</STPCHKSYNCRQ>		

11.12.1.2 Response <STPCHKSYNCRS>

Tag	Version	Description
<STPCHKSYNCRS>		Synchronization-response aggregate
<TOKEN>	V1 only	New synchronization token, ***token***
<TOKEN2>	V2 only	New synchronization token, ***token2***
<LOSTSYNC>		Y if the token in the synchronization request is older than the earliest entry in the server's history table. In this case, some responses have been lost. N if the token in the synchronization request is newer than or matches a token in the server's history table. ***Boolean***
<SYNCERROR>	V2 only	Optional error code, ***N-6***
<BANKACCTFROM>		Bank account of interest; token must be interpreted in terms of this account
</BANKACCTFROM>		
<STPCHKTRNRS>		Stop-check transactions (0 or more)
</STPCHKTRNRS>		
</STPCHKSYNCRS>		

11.12.2 Data Synchronization for Intrabank Funds Transfers

11.12.2.1 Request <INTRASYNCRQ>

Tag	Version	Description
<INTRASYNCRQ>		Synchronization-request aggregate
Client synchronization option; <TOKEN>, <TOKEN2>, <TOKENONLY>, or <REFRESH>		
<TOKEN>	V1 only	Previous value of <TOKEN> received for this type of synchronization request from server; 0 for first-time

Tag	Version	Description
		requests; ***token***
<TOKEN2>	V2 only	Previous value of <TOKEN2> received for this type of synchronization request from server; 0 for first-time requests; ***token2***
<TOKENONLY>		Request for just the current <TOKEN> without the history, ***Boolean***
<REFRESH>		Request for refresh of current state, ***Boolean***
<REJECTIFMISSING>		If Y, do not process requests if client <TOKEN> is out of date, ***Boolean***
Account-from options. Choose either <BANKACCTFROM> or <CCACCTFROM>.		
<BANKACCTFROM>		Account-from aggregate, see section 11.3.1
</BANKACCTFROM>		
-or-		
<CCACCTFROM>	V2 only	Credit-card-account-from aggregate, see section 11.3.2
</CCACCTFROM>		
<INTRATRNRQ>		Intrabank-funds-transfer transactions (0 or more)
</INTRATRNRQ>		
</INTRASYNCRQ>		

11.12.2.2 Response <INTRASYNCRS>

Tag	Version	Description
<INTRASYNCRS>		Synchronization-response aggregate
<TOKEN>	V1 only	New synchronization token, ***token***
<TOKEN2>	V2 only	New synchronization token, ***token2***
<LOSTSYNC>		Y if the token in the synchronization request is older than the earliest entry in the server's history table. In this case, some responses have been lost. N if the token in the synchronization request is newer than or matches a token in the server's history table. ***Boolean***
<SYNCERROR>	V2 only	Optional error code, ***N-6***
Account-from options. Choose either <BANKACCTFROM> or <CCACCTFROM>.		
<BANKACCTFROM>		Account-from aggregate, see section 11.3.1
</BANKACCTFROM>		
-or-		
<CCACCTFROM>	V2 only	Credit-card-account-from aggregate, see section 11.3.2
</CCACCTFROM>		
<INTRATRNRS>		Intrabank-funds-transfer transactions (0 or more)
</INTRATRNRS>		
</INTRASYNCRS>		

The <INTRASYNCRS> responses contain only intrabank transfers where the BANKACCTFROM or CCACCTFROM matches that submitted in the sync request.

11.12.3 Data Synchronization for Interbank Funds Transfers

11.12.3.1 Request <INTERSYNCRQ>

Tag	*Version*	*Description*
<INTERSYNCRQ>		Synchronization-request aggregate
Client synchronization option; <TOKEN>, <TOKEN2>, <TOKENONLY>, or <REFRESH>		
<TOKEN>	V1 only	Previous value of <TOKEN> received for this type of synchronization request from server; 0 for first-time requests; ***token***
<TOKEN2>	V2 only	Previous value of <TOKEN2> received for this type of synchronization request from server; 0 for first-time requests; ***token2***
<TOKENONLY>		Request for just the current <TOKEN> without the history, ***Boolean***
<REFRESH>		Request for refresh of current state, ***Boolean***
<REJECTIFMISSING>		If Y, do not process requests if client <TOKEN> is out of date, ***Boolean***
Account-from options. Choose either <BANKACCTFROM> or <CCACCTFROM>.		
<BANKACCTFROM>		Account-from aggregate, see section 11.3.1
</BANKACCTFROM>		
-or-		
<CCACCTFROM>	V2 only	Credit-card-account-from aggregate, see section 11.3.2
</CCACCTFROM>		
<INTERTRNRQ>		Interbank-funds-transfer transactions (0 or more)
</INTERTRNRQ>		
<MULTIINTERTRNRQ>	V2 only	Multiple interbank-funds-transfer transactions (0 or more)
</MULTIINTERTRNRQ>		
</INTERSYNCRQ>		

11.12.3.2 Response <INTERSYNCRS>

Tag	*Version*	*Description*
<INTERSYNCRS>		Synchronization-response aggregate
<TOKEN>	V1 only	New synchronization token, ***token***
<TOKEN2>	V2 only	New synchronization token, ***token2***
<LOSTSYNC>		Y if the token in the synchronization request is older than the earliest entry in the server's history table. In this case, some responses have been lost. N if the token in the synchronization request is newer than or matches a token in the server's history table. ***Boolean***
<SYNCERROR>	V2 only	Optional error code, ***N-6***
Account-from options. Choose either		

Tag	Version	Description
<BANKACCTFROM> or <CCACCTFROM>.		
<BANKACCTFROM>		Account-from aggregate, see section 11.3.1
</BANKACCTFROM>		
-or-		
<CCACCTFROM>	V2 only	Credit-card-account-from aggregate, see section 11.3.2
</CCACCTFROM>		
<INTERTRNRS>		Interbank-funds-transfer transactions (0 or more)
</INTERTRNRS>		
<MULTIINTERTRNRS>	V2 only	Multiple interbank-funds-transfer transactions (0 or more)
</MULTIINTERTRNRS>		
</INTERSYNCRS>		

11.12.4 Data Synchronization for Wire Funds Transfers

11.12.4.1 Request <WIRESYNCRQ>

Tag	Version	Description
<WIRESYNCRQ>		Synchronization-request aggregate
Client synchronization option; <TOKEN>, <TOKEN2>, <TOKENONLY>, or <REFRESH>		
<TOKEN>	V1 only	Previous value of <TOKEN> received for this type of synchronization request from server; 0 for first-time requests; ***token***
<TOKEN2>	V2 only	Previous value of <TOKEN2> received for this type of synchronization request from server; 0 for first-time requests; ***token2***
<TOKENONLY>		Request for just the current <TOKEN> without the history, ***Boolean***
<REFRESH>		Request for refresh of current state, ***Boolean***
<REJECTIFMISSING>		If Y, do not process requests if client <TOKEN> is out of date, ***Boolean***
<BANKACCTFROM>		Bank account of interest; token must be interpreted in terms of this account.
</BANKACCTFROM>		
<WIRETRNRQ>		Wire-transfer transactions (0 or more)
</WIRETRNRQ>		
</WIRESYNCRQ>		

11.12.4.2 Response <WIRESYNCRS>

Tag	Version	Description
<WIRESYNCRS>		Synchronization-response aggregate
<TOKEN>	V1 only	New synchronization token, ***token***
<TOKEN2>	V2 only	New synchronization token, ***token2***

Tag	Version	Description
<LOSTSYNC>		Y if the token in the synchronization request is older than the earliest entry in the server's history table. In this case, some responses have been lost. N if the token in the synchronization request is newer than or matches a token in the server's history table. ***Boolean***
<SYNCERROR>	V2 only	Optional error code, ***N-6***
<BANKACCTFROM>		Bank account of interest; token must be interpreted in terms of this account
</BANKACCTFROM>		
<WIRETRNRS>		Wire-transfer transactions (0 or more)
</WIRETRNRS>		
</WIRESYNCRS>		

11.12.5 Data Synchronization for Recurring Intrabank Funds Transfers

11.12.5.1 Request <RECINTRASYNCRQ>

This request will synchronize the client with the server in relation to recurring intrabank transfer models. To synchronize individual transfers that were created by the model (and perhaps canceled by another client), the client must also issue an <INTRASYNCRQ>.

Tag	Version	Description
<RECINTRASYNCRQ>		Synchronization request
Client synchronization option; <TOKEN>, <TOKEN2>, <TOKENONLY>, or <REFRESH>		
<TOKEN>	V1 only	Previous value of <TOKEN> received for this type of synchronization request from server; 0 for first-time requests; ***token***
<TOKEN2>	V2 only	Previous value of <TOKEN2> received for this type of synchronization request from server; 0 for first-time requests; ***token2***
<TOKENONLY>		Request for just the current <TOKEN> without the history, ***Boolean***
<REFRESH>		Request for refresh of current state, ***Boolean***
<REJECTIFMISSING>		If Y, do not process requests if client <TOKEN> is out of date, ***Boolean***
Account-from options. Choose either <BANKACCTFROM> or <CCACCTFROM>.		
<BANKACCTFROM>		Account-from aggregate, see section 11.3.1
</BANKACCTFROM>		
-or-		
<CCACCTFROM>	V2 only	Credit-card-account-from aggregate, see section 11.3.2
</CCACCTFROM>		
<RECINTRATRNRQ>		Recurring-intrabank-funds-transfer transactions (0 or more)
</RECINTRATRNRQ>		

Tag	Version	Description
</RECINTRASYNCRQ>		

11.12.5.2 Response <RECINTRASYNCRS>

Tag	Version	Description
<RECINTRASYNCRS>		Synchronization-response aggregate
<TOKEN>	V1 only	New synchronization token, ***token***
<TOKEN2>	V2 only	New synchronization token, ***token2***
<LOSTSYNC>		Y if the token in the synchronization request is older than the earliest entry in the server's history table. In this case, some responses have been lost. N if the token in the synchronization request is newer than or matches a token in the server's history table. ***Boolean***
<SYNCERROR>	V2 only	Optional error code, ***N-6***
Account-from options. Choose either <BANKACCTFROM> or <CCACCTFROM>.		
<BANKACCTFROM>		Account-from aggregate, see section 11.3.1
</BANKACCTFROM>		
-or-		
<CCACCTFROM>	V2 only	Credit-card-account-from aggregate, see section 11.3.2
</CCACCTFROM>		
<RECINTRATRNRS>		Recurring-intrabank-funds-transfer transactions (0 or more)
</RECINTRATRNRS>		
</RECINTRASYNCRS>		

The <RECINTRASYNCRS> responses contain only intrabank transfer models where the BANKACCTFROM or CCACCTFROM matches that submitted in the sync request.

11.12.6 Data Synchronization for Recurring Interbank Funds Transfers

11.12.6.1 Request <RECINTERSYNCRQ>

This request will synchronize the client with the server in relation to recurring interbank transfer models. To synchronize individual funds transfers that were created by the model (and perhaps canceled by another client), the client must also issue an <INTERSYNCRQ>.

Tag	Version	Description
<RECINTERSYNCRQ>		Synchronization-request aggregate
Client synchronization option; <TOKEN>, <TOKEN2>, <TOKENONLY>, or <REFRESH>		
<TOKEN>	V1 only	Previous value of <TOKEN> received for this type of synchronization request from server; 0 for first-time requests; ***token***

Tag	Version	Description
<TOKEN2>	V2 only	Previous value of <TOKEN2> received for this type of synchronization request from server; 0 for first-time requests; ***token2***
<TOKENONLY>		Request for just the current <TOKEN> without the history, ***Boolean***
<REFRESH>		Request for refresh of current state, ***Boolean***
<REJECTIFMISSING>		If Y, do not process requests if client <TOKEN> is out of date, ***Boolean***
Account-from options. Choose either <BANKACCTFROM> or <CCACCTFROM>.		
<BANKACCTFROM>		Account-from aggregate, see section 11.3.1
</BANKACCTFROM>		
-or-		
<CCACCTFROM>	V2 only	Credit-card-account-from aggregate, see section 11.3.2
</CCACCTFROM>		
<RECINTERTRNRQ>		Recurring-transfer transactions (0 or more)
</RECINTERTRNRQ>		
</RECINTERSYNCRQ>		

11.12.6.2 Response <RECINTERSYNCRS>

Tag	Version	Description
<RECINTERSYNCRS>		Synchronization-response aggregate
<TOKEN>	V1 only	New synchronization token, ***token***
<TOKEN2>	V2 only	New synchronization token, ***token***
<LOSTSYNC>		Y if the token in the synchronization request is older than the earliest entry in the server's history table. In this case, some responses have been lost. N if the token in the synchronization request is newer than or matches a token in the server's history table. ***Boolean***
<SYNCERROR>	V2 only	Optional error code, ***N-6***
Account-from options. Choose either <BANKACCTFROM> or <CCACCTFROM>.		
<BANKACCTFROM>		Account-from aggregate, see section 11.3.1
</BANKACCTFROM>		
-or-		
<CCACCTFROM>	V2 only	Credit-card-account-from aggregate, see section 11.3.2
</CCACCTFROM>		
<RECINTERTRNRS>		Recurring-interbank-funds-transfer transactions (0 or more)
</RECINTERTRNRS>		
</RECINTERSYNCRS>		

11.12.7 Data Synchronization for Bank Mail

11.12.7.1 Request <BANKMAILSYNCRQ>

Tag	*Version*	*Description*
<BANKMAILSYNCRQ>		Synchronization-request aggregate
Client synchronization option; <TOKEN>, <TOKEN2>, <TOKENONLY>, or <REFRESH>		
<TOKEN>	V1 only	Previous value of <TOKEN> received for this type of synchronization request from server; 0 for first-time requests; ***token***
<TOKEN2>	V2 only	Previous value of <TOKEN2> received for this type of synchronization request from server; 0 for first-time requests; ***token2***
<TOKENONLY>		Request for just the current <TOKEN> without the history, ***Boolean***
REFRESH>		Request for refresh of current state, ***Boolean***
<REJECTIFMISSING>		If Y, do not process requests if client <TOKEN> is out of date, ***Boolean***
<INCIMAGES>		Y if the client accepts mail with images in the message body. N if the client does not accept mail with images in the message body. ***Boolean***
<USEHTML>		Y if client wants an HTML response, N if client wants plain text, ***Boolean***
Account-from options. Choose either <BANKACCTFROM> or <CCACCTFROM>.		
<BANKACCTFROM>		Account-from aggregate, see section 11.3.1
</BANKACCTFROM>		
-or-		
<CCACCTFROM>		Credit-card-account-from aggregate, see section 11.3.2
</CCACCTFROM>		
<BANKMAILTRNRQ>		Bank-mail transactions (0 or more)
</BANKMAILTRNRQ>		
</BANKMAILSYNCRQ>		

11.12.7.2 Response <BANKMAILSYNCRS>

Tag	*Version*	*Description*
<BANKMAILSYNCRS>		Synchronization-response aggregate
<TOKEN>	V1 only	New synchronization token, ***token***
<TOKEN2>	V2 only	New synchronization token, ***token2***
<LOSTSYNC>		Y if the token in the synchronization request is older than the earliest entry in the server's history table. In this case, some responses have been lost. N if the token in the synchronization request is newer than or matches a token in the server's history table. ***Boolean***
<SYNCERROR>	V2 only	Optional error code, ***N-6***
Account-from options. Choose either <BANKACCTFROM> or <CCACCTFROM>.		
<BANKACCTFROM>		Account-from aggregate, see section 11.3.1

Tag	Version	Description
</BANKACCTFROM>		
-or-		
<CCACCTFROM>		Credit-card-account-from aggregate, see section 11.3.2
</CCACCTFROM>		
<BANKMAILTRNRS>		Bank-mail transactions (0 or more)
</BANKMAILTRNRS>		
</BANKMAILSYNCRS>		

11.12.8 Status Codes

All synchronization responses can return the following status codes based on the <BANKACCTFROM> in the synchronization request:

Code	Meaning
0	Success (INFO)
2000	General error (ERROR)
2002	General account error (ERROR)
2003	Account not found (ERROR)
2004	Account closed (ERROR)
2005	Account not authorized (ERROR)

11.13 Message Sets and Profile

Open Financial Exchange separates messages that the client and server send into groups called message sets. Each FI defines the message sets that the institution supports. The messages described in this section fall into the following types:

- Banking – includes statement download, closing statement download, bank e-mail, notification, and intrabank funds transfer
- Credit Card – credit card statement download and closing statement download
- Interbank Funds Transfers
- Wire Funds Transfers

Each message set contains options and attributes that allow an FI to customize its use of Open Financial Exchange. For example, an institution can support the Interbank Funds Transfer Message Set (INTERXFERMSGSETV1), but it can choose not to support the recurring form of these transfers.

The profile defines the options and attributes as part of each message-set definition. Each set of options and attributes appears within an aggregate that is specific to a message set. For example, <WIREXFERMSGSETV1> contains all of the options and attributes that pertain to wire transfers.

11.13.1 Message Sets and Messages

11.13.1.1 Bank Message Set and Messages

Note: *BANKMSGSET version revision due to adding CCACCTFROM as a funding account to the INTRASYNCRQ/S and the REINTRASYNCRQ/S*

11.13.1.1.1 Bank Message Set Request Messages

Message Set	*Message*
<BANKMSGSET>	
<BANKMSGSETV1>	
<BANKMSGSRQV1>	STMTTRNRQ
	STMTRQ
	STMTENDTRNRQ
	STMTENDRQ
	STPCHKTRNRQ
	STPCHKRQ
	INTRATRNRQ
	INTRARQ
	INTRAMODRQ
	INTRACANRQ
	RECINTRATRNRQ
	RECINTRARQ
	RECINTRAMODRQ
	RECINTRACANRQ
	BANKMAILTRNRQ
	BANKMAILRQ
	STPCHKSYNCRQ
	INTRASYNCRQ
	RECINTRASYNCRQ
	BANKMAILSYNCRQ
</BANKMSGSRQV1>	
</BANKMSGSETV1>	
<BANKMSGSETV2>	
<BANKMSGSRQV2>	STMTTRNRQ
	STMTRQ
	STMTENDTRNRQ
	STMTENDRQ
	STPCHKTRNRQ
	STPCHKRQ
	INTRATRNRQ
	INTRARQ
	INTRAMODRQ
	INTRACANRQ
	RECINTRATRNRQ
	RECINTRARQ
	RECINTRAMODRQ
	RECINTRACANRQ
	BANKMAILTRNRQ

Message Set	Message
	BANKMAILRQ
	STPCHKSYNCRQ
	INTRASYNCRQ
	RECINTRASYNCRQ
	BANKMAILSYNCRQ
</BANKMSGSRQV2>	
</BANKMSGSETV2>	
</BANKMSGSET>	

11.13.1.1.2 Bank Message Set Response Messages

Message Set	Message
<BANKMSGSET>	
<BANKMSGSETV1>	
<BANKMSGSRSV1>	STMTTRNRS
	STMTRS
	STMTENDTRNRS
	STMTENDRS
	STPCHKTRNRS
	STPCHKRS
	INTRATRNRS
	INTRARS
	INTRAMODRS
	INTRACANRS
	RECINTRATRNRS
	RECINTRARS
	RECINTRAMODRS
	RECINTRACANRS
	BANKMAILTRNRS
	BANKMAILRS
	CHKMAILRS
	DEPMAILRS
	STPCHKSYNCRS
	INTRASYNCRS
	RECINTRASYNCRS
	BANKMAILSYNCRS
</BANKMSGSRSV1>	
</BANKMSGSETV1>	
<BANKMSGSETV2>	
<BANKMSGSRSV2>	STMTTRNRS
	STMTRS

Message Set	Message
	STMTENDTRNRS
	STMTENDRS
	STPCHKTRNRS
	STPCHKRS
	INTRATRNRS
	INTRARS
	INTRAMODRS
	INTRACANRS
	RECINTRATRNRS
	RECINTRARS
	RECINTRAMODRS
	RECINTRACANRS
	BANKMAILTRNRS
	BANKMAILRS
	CHKMAILRS
	DEPMAILRS
	STPCHKSYNCRS
	INTRASYNCRS
	RECINTRASYNCRS
	BANKMAILSYNCRS
</BANKMSGSRSV2>	
</BANKMSGSETV2>	
</BANKMSGSET>	

11.13.1.2 Credit Card Message Set and Messages

11.13.1.2.1 Credit Card Message Set Request Messages

Message Set	Message
<CREDITCARDMSGSET>	
<CREDITCARDMSGSETV1>	
<CREDITCARDMSGSRQV1>	CCSTMTTRNRQ
	CCSTMTRQ
	CCSTMTENDTRNRQ
	CCSTMTENDRQ
</CREDITCARDMSGSRQV1>	
</CREDITCARDMSGSETV1>	
<CREDITCARDMSGSETV2>	
<CREDITCARDMSGSRQV2>	CCSTMTTRNRQ
	CCSTMTRQ
	CCSTMTENDTRNRQ
	CCSTMTENDRQ

Message Set	Message
</CREDITCARDMSGSRQV2>	
</CREDITCARDMSGSETV2>	
</CREDITCARDMSGSET>	

11.13.1.2.2 Credit Card Message Set Response Messages

Message Set	Message
<CREDITCARDMSGSET>	
<CREDITCARDMSGSETV1>	
<CREDITCARDMSGSRSV1>	CCSTMTTRNRS
	CCSTMTRS
	CCSTMTENDTRNRS
	CCSTMTENDRS
</CREDITCARDMSGSRSV1>	
</CREDITCARDMSGSETV1>	
<CREDITCARDMSGSETV2>	
<CREDITCARDMSGSRSV2>	CCSTMTTRNRS
	CCSTMTRS
	CCSTMTENDTRNRS
	CCSTMTENDRS
</CREDITCARDMSGSRSV2>	
</CREDITCARDMSGSETV2>	
</CREDITCARDMSGSET>	

11.13.1.3 Interbank Transfer Message Set and Messages

11.13.1.3.1 Interbank Transfer Message Set Request Messages

Message Set	Message
<INTERXFERMSGSET>	
<INTERXFERMSGSETV1>	
<INTERXFERMSGSRQV1>	INTERTRNRQ
	INTERRQ
	INTERMODRQ
	INTERCANRQ
	RECINTERTRNRQ
	RECINTERRQ
	RECINTERMODRQ
	RECINTERCANRQ
	INTERSYNCRQ
	RECINTERSYNCRQ
</INTERXFERMSGSRQV1>	

Message Set	Message
</INTERXFERMSGSETV1>	
<INTERXFERMSGSETV2>	
<INTERXFERMSGSRQV2>	INTERTRNRQ
	INTERRQ
	INTERMODRQ
	INTERCANRQ
	MULTIINTERTRNRQ
	INTERRQ
	RECINTERTRNRQ
	RECINTERRQ
	RECINTERMODRQ
	RECINTERCANRQ
	INTERSYNCRQ
	RECINTERSYNCRQ
</INTERXFERMSGSRQV2>	
</INTERXFERMSGSETV2>	
</INTERXFERMSGSET>	

11.13.1.3.2 Interbank Transfer Message Set Response Messages

Message Set	Message
<INTERXFERMSGSET>	
<INTERXFERMSGSETV1>	
<INTERXFERMSGSETRSV1>	INTERTRNRS
	INTERRS
	INTERMODRS
	INTERCANRS
	RECINTERTRNRS
	RECINTERRS
	RECINTERMODRS
	RECINTERCANRS
	INTERSYNCRS
	RECINTERSYNCRS
</INTERXFERMSGSRSV1>	
</INTERXFERMSGSETV1>	
<INTERXFERMSGSETV2>	
<INTERXFERMSGSETRSV2>	INTERTRNRS
	INTERRS
	INTERMODRS
	INTERCANRS
	MULTIINTERTRNRS

Message Set	Message
	INTERRS
	RECINTERTRNRS
	RECINTERRS
	RECINTERMODRS
	RECINTERCANRS
	INTERSYNCRS
	RECINTERSYNCRS
</INTERXFERMSGSRSV2>	
</INTERXFERMSGSETV2>	
</INTERXFERMSGSET>	

11.13.1.4 Wire Transfer Message Set and Messages

11.13.1.4.1 Wire Transfer Message Set Request Messages

Message Set	Message
<WIREXFERMSGSET>	
<WIREXFERMSGSETV1>	
<WIREXFERMSGSRQV1>	WIRETRNRQ
	WIRERQ
	WIRECANRQ
	WIRESYNCRQ
</WIREXFERMSGSRQV1>	
</WIREXFERMSGSETV1>	
<WIREXFERMSGSETV2>	
<WIREXFERMSGSRQV2>	WIRETRNRQ
	WIRERQ
	WIRECANRQ
	WIRESYNCRQ
</WIREXFERMSGSRQV2>	
</WIREXFERMSGSETV2>	
</WIREXFERMSGSET>	

11.13.1.4.2 Wire Transfer Message Set Response Messages

Message Set	Message
<WIREXFERMSGSET>	
<WIREXFERMSGSETV1>	
<WIREXFERMSGSRSV1>	WIRETRNRS
	WIRERS
	WIRECANRS
	WIRESYNCRS

Message Set	*Message*
</WIREXFERMSGSRSV1>	
</WIREXFERMSGSETV1>	
<WIREXFERMSGSETV2>	
<WIREXFERMSGSRSV2>	WIRETRNRS
	WIRERS
	WIRECANRS
	WIRESYNCRS
</WIREXFERMSGSRSV2>	
</WIREXFERMSGSETV2>	
</WIREXFERMSGSET>	

11.13.2 Bank Message Set Profile

11.13.2.1 <BANKMSGSET>, <BANKMSGSETV1>, <BANKMSGSETV2>

Tag	*Description*
<BANKMSGSET>	Message set for banking
<BANKMSGSETV1>	Version 1 of message set
<MSGSETCORE>	Common message-set core
</MSGSETCORE>	
<INVALIDACCTTYPE>	Account type not supported in <BANKACCTFROM>; 0 or more of account types, see section 11.3.1.2 for values
<CLOSINGAVAIL>	Closing statement information available, ***Boolean***
<XFERPROF>	Intrabank transfer profile (if supported), see section 11.13.2.2
</XFERPROF>	
<STPCHKPROF>	Stop check profile (if supported), see section 11.13.2.3
</STPCHKPROF>	
<EMAILPROF>	E-mail profile, see section 11.13.2.4
</EMAILPROF>	
</BANKMSGSETV1>	End of bank message set version 1
<BANKMSGSETV2>	Version 2 of message set
<MSGSETCORE>	Common message-set core
</MSGSETCORE>	
<INVALIDACCTTYPE2>	Account type not supported in <BANKACCTFROM>; 0 or more of account types, see section 11.3.1.2 for values
<CLOSINGAVAIL>	Closing statement information available, ***Boolean***
<XFERPROF>	Intrabank transfer profile (if supported), see section 11.13.2.2
</XFERPROF>	
<STPCHKPROF>	Stop check profile (if supported), see section 11.13.2.3
</STPCHKPROF>	

Tag	Description
<EMAILPROF>	E-mail profile, see section 11.13.2.4
</EMAILPROF>	
</BANKMSGSETV2>	End of bank message set version 2
</BANKMSGSET>	

11.13.2.2 Banking Profile, Funds Transfer <XFERPROF>

Tag	Version	Description
<XFERPROF>		Intrabank transfer profile (if supported)
<PROCDAYSOFF>		Days of week that no processing occurs: MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, SATURDAY, or SUNDAY. 0 or more <PROCDAYSOFF> can be sent.
<PROCENDTM>		Time of day that day's processing ends, ***time***
<CANSCHED>		Supports scheduled transfers, ***Boolean***
<CANRECUR>		Supports recurring transfers, ***Boolean***. Requires <CANSCHED>
<CANMODXFERS>		Permit modifications to transfers, i.e. <INTRAMODRQ>, ***Boolean***
<CANMODMDLS>		Permit modifications to models, i.e. <RECINTRAMODRQ>, ***Boolean***
<MODELWND>		Model window; the number of days before a recurring transaction is scheduled to be processed that it is instantiated on the system, ***N-3***
<DAYSWITH>		Number of days before processing date that funds are withdrawn, ***N-3***
<DFLTDAYSTOPAY>		Default number of days to pay, ***N-3***
<NEEDTANTRANSFER>	V2 only	A TAN is required for user authentication in the transaction wrapper of a request to create, modify or cancel an intrabank transfer (see section 2.4.6), ***Boolean***
		Not used in USA
<SUPPORTDTAVAIL>	V2 only	Support for specifying a transfer value date using the DTAVAIL tag in the <XFERINFO> aggregate (see section 11.3.5), ***Boolean***
		Not used in USA
</XFERPROF>		

11.13.2.3 Banking Profile, Stop Checks <STPCHKPROF>

Tag	Description
<STPCHKPROF>	Stop check profile (if supported)
<PROCDAYSOFF>	Days of week that no processing occurs: MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, SATURDAY, or SUNDAY. 0 or more <PROCDAYSOFF> can be sent.
<PROCENDTM>	Time of day that day's processing ends, ***time***
<CANUSERANGE>	Can stop a range of checks, ***Boolean.***
<CANUSEDESC>	Can stop by description, ***Boolean.***

Tag	Description
<**STPCHKFEE**>	Default stop check free ***Amount***
</STPCHKPROF>	

11.13.2.4 Banking Profile, Email <EMAILPROF>

Tag	Description
<EMAILPROF>	E-mail profile
<CANEMAIL>	Supports generalized banking e-mail, ***Boolean***
<CANNOTIFY>	Supports notification (of any kind), ***Boolean***
</EMAILPROF>	

11.13.3 Credit Card Message Set Profile

Tag	Description
<CREDITCARDMSGSET>	Beginning tag for credit card message set
<CREDITCARDMSGSETV1>	Version 1 of message set
<MSGSETCORE>	Common message-set core
</MSGSETCORE>	
<CLOSINGAVAIL>	Closing statement information available, ***Boolean***
</CREDITCARDMSGSETV1>	Ending tag of credit card message set version 1
<CREDITCARDMSGSETV2>	Version 2 of message set
<MSGSETCORE>	Common message-set core
</MSGSETCORE>	
<CLOSINGAVAIL>	Closing statement information available, ***Boolean***
</CREDITCARDMSGSETV2>	Ending tag of credit card message set version 2
</CREDITCARDMSGSET>	Ending tag of credit card message set

11.13.4 Interbank Funds Transfer Message Set Profile

Tag	Description
<INTERXFERMSGSET>	Beginning tag for interbank transfers message set
<INTERXFERMSGSETV1>	Version 1 of message set
<MSGSETCORE>	Common message-set core
</MSGSETCORE>	
<XFERPROF>	Interbank transfer profile, same as XFERPROF in banking, see section 11.13.2.2
</XFERPROF>	
<CANBILLPAY>	Server is capable of handling bill payment as a form of transfers, ***Boolean***
<CANCELWND>	Number of days after an interbank transfer occurs that it can be canceled, ***N-3***
<DOMXFERFEE>	Standard fee for a domestic interbank transfer, ***Amount***
<INTLXFERFEE>	Standard fee for an international interbank transfer, ***Amount***
</INTERXFERMSGSETV1>	End of interbank transfer message set version 1
<INTERXFERMSGSETV2>	Version 2 of message set

Tag	Description
<MSGSETCORE>	Common message-set core
</MSGSETCORE>	
<XFERPROF>	Interbank transfer profile, same as XFERPROF in banking, see section 11.13.2.2
</XFERPROF>	
<CANBILLPAY>	Server is capable of handling bill payment as a form of transfers, ***Boolean***
<CANCELWND>	Number of days after an interbank transfer occurs that it can be canceled, ***N-3***
<DOMXFERFEE>	Standard fee for a domestic interbank transfer, ***Amount***
<INTLXFERFEE>	Standard fee for an international interbank transfer, ***Amount***
<CANMULTI>	Server is capable of handling multiple interbank funds transfer transactions, <MULTIINTERTRNRQ>.,***Boolean***
</INTERXFERMSGSETV2>	End of interbank transfer message set version 2
</INTERXFERMSGSET>	End of interbank transfer message set

11.13.5 Wire Transfer Message Set Profile

Tag	Description
<WIREXFERMSGSET>	Core message set for wire transfers
<WIREXFERMSGSETV1>	Version 1 of message set
<MSGSETCORE>	Common message-set core
</MSGSETCORE>	
<PROCDAYSOFF>	Days of week that no processing occurs: MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, SATURDAY, or SUNDAY. 0 or more <PROCDAYSOFF> can be sent.
<PROCENDTM>	Time of day that day's processing ends, ***time***
<CANSCHED>	Supports scheduled transfers, ***Boolean***
<DOMXFERFEE>	Standard fee for a domestic wire transfer, ***Amount***
<INTLXFERFEE>	Standard fee for an international wire transfer, ***Amount***
</WIREXFERMSGSETV1>	End of wire transfer message set version 1
<WIREXFERMSGSETV2>	Version 2 of message set
<MSGSETCORE>	Common message-set core
</MSGSETCORE>	
<PROCDAYSOFF>	Days of week that no processing occurs: MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, SATURDAY, or SUNDAY. 0 or more <PROCDAYSOFF> can be sent.
<PROCENDTM>	Time of day that day's processing ends, ***time***
<CANSCHED>	Supports scheduled transfers, ***Boolean***
<DOMXFERFEE>	Standard fee for a domestic wire transfer, ***Amount***
<INTLXFERFEE>	Standard fee for an international wire transfer, ***Amount***
</WIREXFERMSGSETV2>	End of wire transfer message set version 2
</WIREXFERMSGSET>	Ending tag of wire transfer message set

11.14 Examples

11.14.1 Statement Download

This example represents a customer who requests a statement download for a checking account. The request omits <DTSTART> and <DTEND> because the client is interested in getting all available data. The response contains an updated balance for the account and two transactions.

The request file:

```
<OFX>                                       <!-- Begin request data -->
   <SIGNONMSGSRQV1>
      <SONRQ>                               <!-- Begin signon -->
         <DTCLIENT>19961029101000           <!-- Oct. 29, 1996, 10:10:00 am -->
         <USERID>123-45-6789                <!-- User ID (that is, SSN) -->
         <USERPASS>MyPassword               <!-- Password (SSL encrypts whole) -->
         <LANGUAGE>ENG                      <!-- Language used for text -->
         <FI>                               <!-- ID of receiving institution -->
            <ORG>NCH <!-- Name of ID owner -->
            <FID>1001                       <!-- Actual ID -->
         </FI>
         <APPID>MyApp
         <APPVER>0500
      </SONRQ>      <!-- End of signon -->
   </SIGNONMSGSRQV1>

   <BANKMSGSRQV1>
      <STMTTRNRQ>   <!-- First request in file -->
         <TRNUID>1001
         <STMTRQ>   <!-- Begin statement request -->
            <BANKACCTFROM>                  <!-- Identify the account -->
               <BANKID>121099999            <!-- Routing transit or other FI ID -->
               <ACCTID>999988               <!-- Account number -->
               <ACCTTYPE>CHECKING           <!-- Account type -->
            </BANKACCTFROM>                 <!-- End of account ID -->
            <INCTRAN>                       <!-- Begin include transaction -->
               <INCLUDE>Y                   <!-- Include transactions -->
            </INCTRAN>                      <!-- End of include transaction -->
         </STMTRQ>                          <!-- End of statement request -->
      </STMTTRNRQ>                          <!-- End of first request -->
   </BANKMSGSRQV1>
</OFX>                                      <!-- End of request data -->
```

The response file:

```
<OFX>                                       <!-- Begin response data -->
   <SIGNONMSGSRSV1>
      <SONRS>                               <!-- Begin signon -->
         <STATUS>   <!-- Begin status aggregate -->
            <CODE>0                         <!-- OK -->
            <SEVERITY>INFO
         </STATUS>
         <DTSERVER>19961029101003           <!-- Oct. 29, 1996, 10:10:03 am -->
         <LANGUAGE>ENG                      <!-- Language used in response -->
         <DTPROFUP>19961029101003           <!-- Last update to profile -->
         <DTACCTUP>19961029101003           <!-- Last account update -->
      </SONRS>      <!-- End of signon -->
   </SIGNONMSGSRSV1>

   <BANKMSGSRSV1>
      <STMTTRNRS>   <!-- First response -->
         <TRNUID>1001                       <!-- Client ID sent in request -->
         <STATUS>   <!-- Start status aggregate -->
```

```
            <CODE>0                         <!-- OK -->
            <SEVERITY>INFO
         </STATUS>
         <STMTRS>    <!-- Begin statement response -->
            <CURDEF>USD
            <BANKACCTFROM>                  <!-- Identify the account -->
               <BANKID>121099999            <!-- Routing transit or other FI ID -->
               <ACCTID>999988               <!-- Account number -->
               <ACCTTYPE>CHECKING           <!-- Account type -->
            </BANKACCTFROM>                 <!-- End of account ID -->
            <BANKTRANLIST>                  <!-- Begin list of statement trans. -->
               <DTSTART>19961001            <!-- Start date: Oct. 1, 1996 -->
               <DTEND>19961028              <!-- End date: Oct. 28, 1996 -->
               <STMTTRN>                    <!-- First statement transaction -->
                  <TRNTYPE>CHECK            <!-- Check -->
                  <DTPOSTED>19961004        <!-- Posted on Oct. 4, 1996 -->
                  <TRNAMT>200.00            <!-- $200.00 -->
                  <FITID>00002              <!-- Unique ID -->
                  <CHECKNUM>1000            <!-- Check number -->
               </STMTTRN>                   <!-- End statement transaction -->
               <STMTTRN>                    <!-- Second transaction -->
               <TRNTYPE>ATM                 <!-- ATM transaction -->
               <DTPOSTED>19961020           <!-- Posted on Oct. 20, 1996 -->
               <DTUSER>19961020             <!-- User date of Oct. 20, 1996 -->
               <TRNAMT>300.00               <!-- $300.00 -->
               <FITID>00003                 <!-- Unique ID -->
            </STMTTRN>                      <!-- End statement transaction -->
            </BANKTRANLIST>                 <!-- End list of statement trans. -->
            <LEDGERBAL>                     <!-- Ledger balance aggregate -->
               <BALAMT>200.29               <!-- Bal amount: $200.29 -->
               <DTASOF>199610291120         <!-- Bal date: 10/29/96, 11:20 am -->
            </LEDGERBAL>                    <!-- End ledger balance -->
            <AVAILBAL>                      <!-- Available balance aggregate -->
               <BALAMT>200.29               <!-- Bal amount: $200.29 -->
               <DTASOF>199610291120         <!-- Bal date: 10/29/96, 11:20 am -->
            </AVAILBAL>                     <!-- End available balance -->
         </STMTRS>                          <!-- End statement response -->
      </STMTTRNRS>                          <!-- End of transaction -->
   </BANKMSGSRSV1>
</OFX>                                      <!-- End of response data -->
```

11.14.2 Intrabank Funds Transfer

This example is for a customer who requests an immediate funds transfer of $200.00 from a checking account to a savings account.

The request file:

```
<OFX>                                       <!-- Begin request data -->
   <SIGNONMSGSRQV1>
      <SONRQ>                               <!-- Begin signon -->
         <DTCLIENT>19960828101000           <!-- Aug 28, 1996, 10:10:00 am -->
         <USERID>123-45-6789                <!-- User ID (e.g. SSN) -->
         <USERPASS>MyPassword               <!-- Password (SSL encrypts whole) -->
         <LANGUAGE>ENG                      <!-- Language used for text -->
         <FI>                               <!-- ID of receiving institution -->
            <ORG>NCH <!-- Name of ID owner -->
            <FID>1001                       <!-- Actual ID -->
         </FI>
         <APPID>MyApp
         <APPVER>0500
      </SONRQ>      <!-- End of signon -->
   </SIGNONMSGSRQV1>
```

```
   <BANKMSGSRQV1>
      <INTRATRNRQ>                        <!-- First request in file -->
         <TRNUID>1001                     <!-- Client's ID for this request -->
         <INTRARQ>                        <!-- Begin transfer request -->
            <XFERINFO>                    <!-- Begin transfer aggregate -->
               <BANKACCTFROM>             <!-- Identify the account -->
                  <BANKID>121099999       <!-- Routing transit or other FI ID -->
                  <ACCTID>999988          <!-- Account number -->
                  <ACCTTYPE>CHECKING      <!-- Account type -->
               </BANKACCTFROM>            <!-- End of account ID -->
               <BANKACCTTO>               <!-- Identify the account -->
                  <BANKID>121099999       <!-- Routing transit or other FI ID -->
                  <ACCTID>999977          <!-- Account number -->
                  <ACCTTYPE>SAVINGS       <!-- Account type -->
               </BANKACCTTO>              <!-- End of account ID -->
               <TRNAMT>200.00             <!-- Amount of transfer -->
            </XFERINFO>                   <!-- End of transfer aggregate -->
         </INTRARQ>                       <!-- End of transfer request -->
      </INTRATRNRQ>                       <!-- End of first request -->
   </BANKMSGSRQV1>
</OFX>                                    <!-- End of request data -->
```

The response file:

```
<OFX>                                     <!-- Begin response data -->
   <SIGNONMSGSRSV1>
      <SONRS>                             <!-- Begin signon -->
         <STATUS>   <!-- Start of status aggregate -->
            <CODE>0                       <!-- OK -->
            <SEVERITY>INFO
         </STATUS>
         <DTSERVER>19960828101003         <!-- Aug 28, 1996, 10:10:03 am -->
         <LANGUAGE>ENG                    <!-- Language used in response -->
         <DTPROFUP>19961029101003         <!-- Last update to profile -->
         <DTACCTUP>19961029101003         <!-- Last account update -->
      </SONRS>      <!-- End of signon -->
   </SIGNONMSGSRSV1>


 <BANKMSGSRSV1>
      <INTRATRNRS>                        <!-- First response in file -->
         <TRNUID>1001                     <!-- Client ID sent in request -->
         <STATUS>   <!-- Start status aggregate -->
            <CODE>0                       <!-- OK -->
            <SEVERITY>INFO
         </STATUS>
         <INTRARS>                        <!-- Begin transfer response -->
            <CURDEF>USD
            <SRVRTID>1001                 <!-- Server assigned ID -->
            <XFERINFO>                    <!-- Begin transfer aggregate -->
               <BANKACCTFROM>             <!-- Identify the account -->
                  <BANKID>121099999       <!-- Routing transit or other FI ID -->
                  <ACCTID>999988          <!-- Account number -->
                  <ACCTTYPE>CHECKING      <!-- Account type -->
               </BANKACCTFROM>            <!-- End of account ID -->
               <BANKACCTTO>               <!-- Identify the account -->
                  <BANKID>121099999       <!-- Routing transit or other FI ID -->
                  <ACCTID>999977          <!-- Account number -->
                  <ACCTTYPE>SAVINGS       <!-- Account type -->
               </BANKACCTTO>              <!-- End of account ID -->
               <TRNAMT>200.00             <!-- Amount of transfer -->
            </XFERINFO>                   <!-- End of transfer aggregate -->
            <DTXFERPRJ>19960829100000     <!-- Projected posting date -->
         </INTRARS>                       <!-- End of transfer response -->
```

```
      </INTRATRNRS>                        <!-- End of first response -->
   </BANKMSGSRSV1>
</OFX>                                     <!-- End of response data -->
```

11.14.3 Stop Check

This example represents a customer who requests a stop for checks 200 through 202. The response indicates that the first check (200) has already posted; the server has stopped the rest of the checks in the range.

The request file:

```
<OFX>                                      <!-- Begin request data -->
   <SIGNONMSGSRQV1>
      <SONRQ>                              <!-- Begin signon -->
         <DTCLIENT>19961029101000          <!-- Oct. 29, 1996, 10:10:00 am -->
         <USERID>123-45-6789               <!-- User ID (e.g. SSN) -->
         <USERPASS>MyPassword              <!-- Password (SSL encrypts whole) -->
         <LANGUAGE>ENG                     <!-- Language used for text -->
         <FI>                              <!-- ID of receiving institution -->
            <ORG>NCH <!-- Name of ID owner -->
            <FID>1001                      <!-- Actual ID -->
         </FI>
         <APPID>MyApp
         <APPVER>0500
      </SONRQ>      <!-- End of signon -->
   </SIGNONMSGSRQV1>

   <BANKMSGSRQV1>
      <STPCHKTRNRQ>                        <!-- First request in file -->
         <TRNUID>1001                      <!-- Client's ID for this request -->
         <STPCHKRQ>                        <!-- Begin stop check request -->
            <BANKACCTFROM>                 <!-- Identify the account -->
               <BANKID>121099999           <!-- Routing transit or other FI ID -->
               <ACCTID>999988              <!-- Account number -->
               <ACCTTYPE>CHECKING          <!-- Account type -->
            </BANKACCTFROM>                <!-- End of account ID -->
            <CHKRANGE>                     <!-- Cancel a range of checks -->
               <CHKNUMSTART>200            <!-- Starting check number -->
               <CHKNUMEND>202              <!-- Ending check number -->
            </CHKRANGE>                    <!-- End range -->
         </STPCHKRQ> <!-- End of stop check request -->
      </STPCHKTRNRQ><!-- End of first request -->
   </BANKMSGSRQV1>
</OFX>                                     <!-- End of request data -->
```

A response file:

```
<OFX>               <!-- Begin response data -->
   <SIGNONMSGSRSV1>
      <SONRS>                              <!-- Begin signon -->
         <STATUS>   <!-- Start status aggregate -->
            <CODE>0                        <!-- OK -->
            <SEVERITY>INFO
         </STATUS>
         <DTSERVER>19961029101003          <!-- Oct. 29, 1996, 10:10:03 am -->
         <LANGUAGE>ENG                     <!-- Language used in response -->
         <DTPROFUP>19961029101003          <!-- Last update to profile -->
         <DTACCTUP>19961029101003          <!-- Last account update -->
      </SONRS>      <!-- End of signon -->
   </SIGNONMSGSRSV1>

   <BANKMSGSRSV1>
      <STPCHKTRNRS>                        <!-- First response in file -->
         <TRNUID>1001                      <!-- Client ID sent in request -->
```

```
         <STATUS>    <!-- Begin status aggregate -->
            <CODE>0                         <!-- OK -->
            <SEVERITY>INFO
         </STATUS>                          <!-- End of status aggregate -->
         <STPCHKRS>                         <!-- Begin stop check response -->
            <CURDEF>USD
            <BANKACCTFROM>                  <!-- Identify the account -->
               <BANKID>121099999            <!-- Routing transit or other FI ID -->
               <ACCTID>999988               <!-- Account number -->
               <ACCTTYPE>CHECKING           <!-- Account type -->
            </BANKACCTFROM>                 <!-- End of account ID -->
            <STPCHKNUM>                     <!-- First stopped check -->
               <CHECKNUM>200                <!-- Check 200 -->
               <CHKSTATUS>101               <!-- Too late - already posted -->
            </STPCHKNUM>                    <!-- End of first stopped check -->
            <STPCHKNUM>                     <!-- Second stopped check -->
               <CHECKNUM>201                <!-- Check 201 -->
               <CHKSTATUS>0                 <!-- OK -->
            </STPCHKNUM>                    <!-- End of second stopped check -->
            <STPCHKNUM>                     <!-- Third stopped check -->
               <CHECKNUM>202                <!-- Check 202 -->
               <CHKSTATUS>0                 <!-- OK -->
            </STPCHKNUM>                    <!-- End of third stopped check -->
            <FEE>10.00
            <FEEMSG>Fee for stop payment.
         </STPCHKRS> <!-- End stop check response -->
      </STPCHKTRNRS><!-- End of transaction -->
   </BANKMSGSRSV1>
</OFX>                                      <!-- End of response data -->
```

11.14.4 Recurring Transfers

This example represents a customer who creates a transfer model and then cancels it. To follow the life of the model (and the transfers it creates), the example includes sessions that occur over a two month period.

The model is added on November 1 and scheduled to start on November 15. The model creates transfers of $1000 from a checking to a savings account. The schedule is open-ended.

The client sends two requests: one to create the model and another to collect any transfers created by the model. The second request is a simple synchronize request.

The request file that creates the model

The client sends the file on November 1:

```
<OFX>                                       <!-- Begin request data -->
   <SIGNONMSGSRQV1>
      <SONRQ>                               <!-- Begin signon -->
         <DTCLIENT>19961101101000           <!-- Nov 1, 1996, 10:10:00 am -->
         <USERID>123-45-6789                <!-- User ID (e.g. SSN) -->
         <USERPASS>MyPassword               <!-- Password (SSL encrypts whole) -->
         <LANGUAGE>ENG                      <!-- Language used for text -->
         <FI>                               <!-- ID of receiving institution -->
            <ORG>NCH <!-- Name of ID owner -->
            <FID>1001                       <!-- Actual ID -->
         </FI>
         <APPID>MyApp
         <APPVER>0500
      </SONRQ>      <!-- End of signon -->
   </SIGNONMSGSRQV1>

   <BANKMSGSRQV1>
      <RECINTRATRNRQ>                       <!-- First request in file -->
         <TRNUID>1001                       <!-- Client's ID for this request -->
```

```
         <RECINTRARQ>                      <!-- Begin request -->
            <RECURRINST>                   <!-- Begin recurring aggregate -->
               <FREQ>MONTHLY               <!-- Monthly schedule -->
            </RECURRINST>                  <!-- End recur aggregate -->
            <INTRARQ>
               <XFERINFO>                  <!-- Begin transfer aggregate -->
                  <BANKACCTFROM>           <!-- Identify the account -->
                     <BANKID>121099999     <!-- Routing transit or other FI ID -->
                     <ACCTID>999988        <!-- Account number -->
                     <ACCTTYPE>CHECKING    <!-- Account type -->
                  </BANKACCTFROM>          <!-- End account ID -->
                  <BANKACCTTO>             <!-- Identify the account -->
                     <BANKID>121099999     <!-- Routing transit or other FI ID -->
                     <ACCTID>999977        <!-- Account number -->
                     <ACCTTYPE>SAVINGS     <!-- Account type -->
                  </BANKACCTTO>            <!-- End of account ID -->
                  <TRNAMT>1000.00          <!-- Amount of transfer      -->
                  <DTDUE>19961115          <!-- First transfer - Nov.15 -->
               </XFERINFO>                 <!-- End transfer aggregate -->
            </INTRARQ>
         </RECINTRARQ>                     <!-- End transfer request -->
      </RECINTRATRNRQ>                     <!-- End first request -->
      <INTRASYNCRQ>                        <!-- Sync intrabank transfers -->
         <TOKEN>0   <!-- Token held by the client -->
         <REJECTIFMISSING>N
            <BANKACCTFROM>                 <!-- Identify the account -->
               <BANKID>121099999
               <ACCTID>999988
               <ACCTTYPE>CHECKING
            </BANKACCTFROM>
      </INTRASYNCRQ><!-- End of sync request -->
   </BANKMSGSRQV1>
</OFX>                                     <!-- End of request data -->
```

The server response provides status for the add recurring transfer request. Assuming that the server creates transfers 30 days prior to posting, and since the client included a funds transfer synchronize request, the server returns status for the first transfer that the model created. This response comes back since the first transfer is scheduled to occur on November 15 and this date falls within 30 days of our session. Had the starting date been more than 30 days from our signon date, the response would have contained only status for the add recurring transfer request.

The response file from the server:

```
<OFX>                                      <!-- Begin response data -->
   <SIGNONMSGSRSV1>
      <SONRS>                              <!-- Begin signon -->
         <STATUS>   <!-- Start of status aggregate-->
            <CODE>0                        <!-- OK -->
            <SEVERITY>INFO
         </STATUS>
         <DTSERVER>19961101101003          <!-- Nov. 1, 1996, 10:10:03 am -->
         <LANGUAGE>ENG                     <!-- Language used in response -->
         <DTPROFUP>19961029101003          <!-- Last update to profile -->
         <DTACCTUP>19961029101003          <!-- Last account update -->
      </SONRS>      <!-- End of signon -->
   </SIGNONMSGSRSV1>

   <BANKMSGSRSV1>
      <RECINTRATRNRS>                      <!-- First response in file -->
         <TRNUID>1001                      <!-- Client ID sent in request -->
         <STATUS>   <!-- Start of status aggregate -->
            <CODE>0                        <!-- OK -->
            <SEVERITY>INFO
         </STATUS>
```

```
      <RECINTRARS>                      <!-- Begin response -->
         <RECSRVRTID>20000              <!-- Server assigned ID -->
         <RECURRINST>                   <!-- Begin recurring aggregate -->
            <FREQ>MONTHLY               <!-- Monthly schedule -->
         </RECURRINST>                  <!-- End of recurring aggregate -->
        <INTRARS>
         <CURDEF>USD
         <SRVRTID>120000
            <XFERINFO>                  <!-- Begin transfer aggregate -->
               <BANKACCTFROM>           <!-- Identify the account -->
                  <BANKID>121099999     <!-- Routing transit or other FI ID -->
                  <ACCTID>999988        <!-- Account number -->
                  <ACCTTYPE>CHECKING    <!-- Account type -->
               </BANKACCTFROM>          <!-- End of account ID -->
               <BANKACCTTO>             <!-- Identify the account -->
                  <BANKID>121099999     <!-- Routing transit or other FI ID -->
                  <ACCTID>999977        <!-- Account number -->
                  <ACCTTYPE>SAVINGS     <!-- Account type -->
               </BANKACCTTO>            <!-- End of account ID -->
               <TRNAMT>1000.00          <!-- Amount of transfer -->
               <DTDUE>19961115          <!-- First transfer - Nov. 15 -->
            </XFERINFO>                 <!-- End of transfer aggregate -->
      </INTRARS>
      </RECINTRARS>                     <!-- End of response -->
   </RECINTRATRNRS>                     <!-- End of first response -->
   <INTRASYNCRS>                        <!-- Sync intrabank transfers -->
      <TOKEN> 22243                     <!-- Token updated -->
      <BANKACCTFROM>                    <!-- Identify the account -->
         <BANKID>121099999
         <ACCTID>999988
         <ACCTTYPE>CHECKING
      </BANKACCTFROM>
      <INTRATRNRS>                      <!-- Second response in file -->
         <TRNUID>1001                   <!-- Client ID sent in request -->
         <STATUS> <!-- Success -->
            <CODE>0                     <!-- OK -->
            <SEVERITY>INFO
         </STATUS>
         <INTRARS>                      <!-- Begin transfer response -->
         <CURDEF>USD
            <SRVRTID>1001               <!-- Server assigned ID -->
            <XFERINFO>                  <!-- Begin transfer aggregate -->
               <BANKACCTFROM>           <!-- Identify the account -->
                  <BANKID>121099999     <!-- Routing transit or other FI ID -->
                  <ACCTID>999988        <!-- Account number -->
                  <ACCTTYPE>CHECKING    <!-- Account type -->
               </BANKACCTFROM>          <!-- End of account ID -->
               <BANKACCTTO>             <!-- Identify the account -->
                  <BANKID>121099999     <!-- Routing transit or other FI ID -->
                  <ACCTID>999977        <!-- Account number -->
                  <ACCTTYPE>SAVINGS     <!-- Account type -->
               </BANKACCTTO>            <!-- End of account ID -->
               <TRNAMT>1000.00          <!-- Amount of transfer -->
            </XFERINFO>                 <!-- End transfer aggregate -->
            <DTXFERPRJ>19961115         <!-- Date (interpret with code) -->
            <RECSRVRTID>20000           <!-- Model that created this xfer -->
         </INTRARS>                     <!-- End of transfer response -->
      </INTRATRNRS>                     <!-- End of second response -->
   </INTRASYNCRS><!-- End of sync response -->
 </BANKMSGSRSV1>
</OFX>                                  <!-- End of response data -->
```

Suppose the customer does not attempt to connect between November 16 and January 1. When the customer does attempt to connect, it is to cancel the recurring transfer model. Since the client sets the <CANPENDING> flag, the cancel is immediate. Any pending transfers are canceled along with the model.

In this example, the client tries to synchronize with the server by requesting any uncollected transfer responses and recurring transfer responses. It does this by sending two synchronization requests. The first is to collect recurring transfer responses. To accomplish this, the recurring transfer cancel request is "wrapped" in a synchronization request. The second synchronization request is to collect individual transfer responses.

The tokens provided in the sync requests tell the server how up to date the client is with respect to recurring transfer models and individual transfers.

The request file:

```
<OFX>                                       <!-- Begin request data -->
   <SIGNONMSGSRQV1>
      <SONRQ>                               <!-- Begin signon -->
         <DTCLIENT>19970101101000           <!-- Jan. 1, 1997, 10:10:00 am -->
         <USERID>123-45-6789                <!-- User ID (e.g. SSN) -->
         <USERPASS>MyPassword               <!-- Password (SSL encrypts whole) -->
         <LANGUAGE>ENG                      <!-- Language used for text -->
         <FI>                               <!-- ID of receiving institution -->
            <ORG>NCH <!-- Name of ID owner -->
            <FID>1001                       <!-- Actual ID -->
         </FI>
         <APPID>MyApp
         <APPVER>0500
      </SONRQ>      <!-- End of signon -->
   </SIGNONMSGSRQV1>

   <BANKMSGSRQV1>
      <RECINTRASYNCRQ>                      <!-- Sync recurring transfers -->
         <TOKEN>324789987                   <!-- Token held by the client -->
         <REJECTIFMISSING>N
         <BANKACCTFROM>
            <BANKID>121099999
            <ACCTID>99998
            <ACCTTYPE>CHECKING
         </BANKACCTFROM>
         <RECINTRATRNRQ>                    <!-- First request in file -->
            <TRNUID>1005                    <!-- Client's ID for this request -->
            <RECINTRACANRQ>                 <!-- Begin recur transfer cancel -->
               <RECSRVRTID>20000            <!-- ID of the model -->
               <CANPENDING>Y                <!-- Cancel pending transfers -->
            </RECINTRACANRQ>                <!-- End recur transfer cancel -->
         </RECINTRATRNRQ>                   <!-- End of first request -->
      </RECINTRASYNCRQ>                     <!-- End of sync request -->
      <INTRASYNCRQ>                         <!-- Sync intrabank transfers -->
         <TOKEN>222289987                   <!-- Token held by the client -->
         <REJECTIFMISSING>N
         <BANKACCTFROM>
            <BANKID>121099999
            <ACCTID>99998
            <ACCTTYPE>CHECKING
         </BANKACCTFROM>
      </INTRASYNCRQ><!-- End of sync request -->
   </BANKMSGSRQV1>
</OFX>                                      <!-- End of request data -->
```

The server responds to this message by canceling the model and by canceling any pending transfers that the model has created. Since the customer last connected, the December 15 transfer has posted, and a transfer has been scheduled for January 15. Since the customer has not connected after November 16, the client has not retrieved the December and January transfer responses. The server will return a total of four transfer responses; two for each of the transfers that the client has not received. This includes two responses for adding the December 15 and January 15 transfers, and two responses for the posting of the December 15 transfer and the cancel of the January 15 transfers.

The response file:

```
<OFX>                                       <!-- Begin response data -->
   <SIGNONMSGSRSV1>
      <SONRS>                               <!-- Begin signon -->
         <STATUS>   <!-- Start of status aggregate -->
            <CODE>0                         <!-- OK -->
            <SEVERITY>INFO
         </STATUS>
         <DTSERVER>19970101101003           <!-- Nov. 1, 1997, 10:10:03 am -->
         <LANGUAGE>ENG                      <!-- Language used in response -->
         <DTPROFUP>19961029101003           <!-- Last update to profile -->
         <DTACCTUP>19961029101003           <!-- Last account update -->
      </SONRS>      <!-- End of signon -->
   </SIGNONMSGSRSV1>

   <BANKMSGSRSV1>
      <RECINTRASYNCRS>                      <!-- Sync response -->
         <TOKEN> 4567888898                 <!-- New token -->
         <BANKACCTFROM>
            <BANKID>121099999
            <ACCTID>99998
            <ACCTTYPE>CHECKING
         </BANKACCTFROM>
         <RECINTRATRNRS>                    <!-- First response in file -->
            <TRNUID>1005                    <!-- Client ID sent in request -->
            <STATUS> <!-- Start of status aggregate -->
               <CODE>0                      <!-- OK -->
               <SEVERITY>INFO
            </STATUS>
            <RECINTRACANRS>                 <!-- Begin cancel model -->
               <RECSRVRTID>20000            <!-- Model that was canceled -->
            <CANPENDING>Y
            </RECINTRACANRS>                <!-- End of cancel model -->
         </RECINTRATRNRS>                   <!-- End of first response -->
      </RECINTRASYNCRS>                     <!-- End of first sync response -->
      <INTRASYNCRS>                         <!-- Sync response for xfers -->
         <TOKEN> 2356789011                 <!-- New token -->
         <BANKACCTFROM>
            <BANKID>121099999
            <ACCTID>99998
            <ACCTTYPE>CHECKING
         </BANKACCTFROM>
         <INTRATRNRS>                       <!-- First response -->
            <TRNUID>1005                    <!-- Client ID sent in request -->
            <STATUS> <!-- Success -->
               <CODE>0                      <!-- OK -->
               <SEVERITY>INFO
            </STATUS>
            <INTRARS>                       <!-- This is the Dec. 15 add -->
             <CURDEF>USD
               <SRVRTID>1003                <!-- Server assigned ID -->
               <XFERINFO>                   <!-- Begin transfer aggregate -->
                  <BANKACCTFROM>            <!-- Identify the account -->
                     <BANKID>121099999      <!-- Routing transit or other FI ID -->
                     <ACCTID>999988         <!-- Account number -->
```

```
            <ACCTTYPE>CHECKING     <!-- Account type -->
         </BANKACCTFROM>           <!-- End of account ID -->
         <BANKACCTTO>              <!-- Identify the account -->
            <BANKID>121099999      <!-- Routing transit or other FI ID -->
            <ACCTID>999977         <!-- Account number -->
            <ACCTTYPE>SAVINGS      <!-- Account type -->
         </BANKACCTTO>             <!-- End of account ID -->
         <TRNAMT>200.00            <!-- Amount of transfer -->
      </XFERINFO>                  <!-- End of transfer aggregate -->
      <DTXFERPRJ>19961215          <!-- Date (interpret with code) -->
      <RECSRVRTID>20000            <!-- Model that was canceled -->
   </INTRARS>                      <!-- End of Dec. 15 transfer -->
</INTRATRNRS>                      <!-- End of first response -->
<INTRATRNRS>                       <!-- Second response -->
   <TRNUID>1005                    <!-- Client ID sent in request -->
   <STATUS>
      <CODE>0                      <!-- OK -->
      <SEVERITY>INFO
   </STATUS>
   <INTRAMODRS>                    <!-- This is the Dec. 15 post -->
      <SRVRTID>1003                <!-- Server assigned ID -->
      <XFERINFO>                   <!-- Begin transfer aggregate -->
         <BANKACCTFROM>            <!-- Identify the account -->
            <BANKID>121099999      <!-- Routing transit or other FI ID -->
            <ACCTID>999988         <!-- Account number -->
            <ACCTTYPE>CHECKING     <!-- Account type -->
         </BANKACCTFROM>           <!-- End of account ID -->
         <BANKACCTTO>              <!-- Identify the account -->
            <BANKID>121099999      <!-- Routing transit or other FI ID -->
            <ACCTID>999977         <!-- Account number -->
            <ACCTTYPE>SAVINGS      <!-- Account type -->
         </BANKACCTTO>             <!-- End of account ID -->
         <TRNAMT>200.00            <!-- Amount of transfer -->
      </XFERINFO>                  <!-- End of transfer aggregate -->
      <XFERPRCSTS>                 <!-- Status of transfer -->
         <XFERPRCCODE>POSTEDON     <!-- Status code -->
         <DTXFERPRC>19961215       <!-- Date (interpret with code) -->
      </XFERPRCSTS>                <!-- End of transfer status -->
   </INTRAMODRS>                   <!-- End of Dec. 15 transfer -->
</INTRATRNRS>                      <!-- End of second response -->
<INTRATRNRS>                       <!-- Third response -->
   <TRNUID>1006                    <!-- Client ID sent in request -->
   <STATUS>
      <CODE>0                      <!-- OK -->
      <SEVERITY>INFO
   </STATUS>
   <INTRARS>                       <!-- This is the Jan. 15 add -->
     <CURDEF>USD
      <SRVRTID>1004                <!-- Server assigned ID -->
      <XFERINFO>                   <!-- Begin transfer aggregate -->
         <BANKACCTFROM>            <!-- Identify the account -->
            <BANKID>121099999      <!-- Routing transit or other FI ID -->
            <ACCTID>999988         <!-- Account number -->
            <ACCTTYPE>CHECKING     <!-- Account type -->
         </BANKACCTFROM>           <!-- End of account ID -->
         <BANKACCTTO>              <!-- Identify the account -->
            <BANKID>121099999      <!-- Routing transit or other FI ID -->
            <ACCTID>999977         <!-- Account number -->
            <ACCTTYPE>SAVINGS      <!-- Account type -->
         </BANKACCTTO>             <!-- End of account ID -->
         <TRNAMT>200.00            <!-- Amount of transfer -->
      </XFERINFO>                  <!-- End of transfer aggregate -->
      <DTXFERPRJ>19970115          <!-- Date of expected posting -->
      <RECSRVRTID>20000            <!-- Model that was canceled -->
```

```
            </INTRARS>                      <!-- End of Jan. 15 transfer -->
         </INTRATRNRS>                      <!-- End of third response -->
         <INTRATRNRS>                       <!-- Fourth response -->
            <TRNUID>1007                    <!-- Client ID sent in this request -->
            <STATUS>
               <CODE>0                      <!-- OK -->
               <SEVERITY>INFO
            </STATUS>
            <INTRACANRS>                    <!-- This is the Jan. 15 cancel -->
               <SRVRTID>1004                <!-- Server ID for Jan. 15 xfer -->
            </INTRACANRS>                   <!-- End of Jan. 15 cancel -->
         </INTRATRNRS>                      <!-- End of fourth xfer response -->
      </INTRASYNCRS><!-- End of second sync response -->
   </BANKMSGSRSV1>
</OFX>                                      <!-- End of response -->
```

12. Payments

This section describes the Payments portion of Open Financial Exchange. Open Financial Exchange Payments consists of a set of functions for scheduling and maintaining payment transactions, and for synchronizing with the server to obtain an accurate status of all recent and scheduled transactions.

Clients use payment requests to schedule payments and to modify or delete payments if necessary. Open Financial Exchange also supports business payments, as described in section 12.1.

The recurring payments function allows the client to schedule automatic generation of a series of recurring payments by means of a single request. As with individual payments, the client can modify or delete these requests.

The payments function incorporates the synchronization features of Open Financial Exchange, allowing multiple client applications to synchronize with the server to obtain the current status of all payment transactions known to the server.

In many international environments, payments are performed using interbank funds transfers. Open Financial Exchange Payments supports this by allowing a payee to be designated as a destination bank account. Servers can implement these messages as transfers where appropriate.

12.1 Consumer and Business Payments

Open Financial Exchange Payments is designed to support both consumer and business payments. Businesses have additional requirements for payments. In particular, there is a need to include itemized instructions that specify how a payment should be disbursed across multiple invoices and/or line items. Open Financial Exchange supports this requirement through the inclusion of the <EXTDPMT> aggregate within payment requests. The Payment Modification Request <PMTMODRQ> also supports changes to <EXTDPMT> data.

12.2 The Payee Model

The payee model in Open Financial Exchange is designed to provide support for both “pay-some” and “pay-any” payment systems. “Pay-some” systems are those that restrict users to only make payments to payees that appear on an approved list. Such payees are often referred to as “standard payees,” or “standard merchants.” These are generally larger corporations that receive high volumes of payments, such as telephone companies or power utilities. In contrast, **“**pay-any” systems allow payments to any payee for which the user provides accurate billing information. These systems often also include a list of standard payees.

12.2.1 Payee Identifiers

Open Financial Exchange is designed to be flexible in the requirements for payee identifiers. It supports systems where all, some, or no payees are assigned a payee ID. In addition, it enables the server to assign an ID to a payee that was previously being paid by billing address.

You must implement the scope of payee such that the ID is at least global across the user’s set of payments-enabled accounts with the payments provider. For example, if the user has both a checking and a money market account enabled for payments with the payments provider, then a payee ID obtained for a payment made from one of these accounts should identify the same payee if used for a payment drawn on the other account. This simplifies client support for allowing a user to choose from which account to make a payment.

Open Financial Exchange requires payee identifiers to have a one-to-one relationship with the corresponding <PAYEE> information. In other words, different payee IDs must also differ in their corresponding payee billing description or payee name <NAME>. Similarly, a payee ID must be independent of a user's account number with the payee. However, the payment system is free to use the user's account number in combination with the payee ID to determine the routing of a payment. These rules are intended to simplify the payee model for the user, insuring that different payee IDs will have discernibly different descriptions associated with them. They also insure that the user will not be required to maintain multiple payee entries for a payee with which the user holds multiple accounts.

Open Financial Exchange includes a tag for indicating the scope of a payee ID returned from the server. This allows clients to adapt by expanding or restricting their functionality depending on the scope of payee IDs it encounters.

A payee list for each user, maintained on the server, allows the server to manage the identifiers assigned to a user's payees. This functionality is described in section 12.2.2.

12.2.2 Payee Lists

Open Financial Exchange specifies that a server-hosted payee list is maintained for each payments user. This list contains the payees that a user has paid through the payment system, or has set up to pay. Updates to this list are available through the synchronization mechanism. This insures that multiple clients have access to the full list of payees the user has configured. It is only necessary to enter each payee once.

Some payment systems require a first time setup before using a payee. This can occur externally to the client and server software, for example by filling out a paper form or telephoning the bank. In this case, payee list synchronization provides a way for the payee to become accessible to the client software when the FI completes the setup.

The list can contain payees with or without payee IDs. An important function of the payee list is to communicate payee changes from the server to the client. This includes changes in processing date parameters and conversion of a payee to a standard payee.

Once added to the list, the payment system makes payments by the payee list ID. This makes it clear to a client when the user is adding to a payee list, and when he or she modifies an existing payee on the list.

Although the messages make it clear whether a client is trying to add a new payee, a careful server will check for exact matches on payee adds and not create new payee list entries unnecessarily.

"Pay-any" systems can perform background processing that matches billing addresses with standard payees. When this occurs, the server can update the relevant payee lists and update the clients when they synchronize with the modified list data.

Each payee entry in the list can also include a list of the user's accounts with that payee. This further reduces the data entry required by a user to make a payment, and facilitates the implementation of lightweight clients.

12.2.3 Standard Payee Lists

Many payment systems maintain a list of payees that receive payments from a large number of users. Payments to these payees are usually consolidated into a few electronic funds transfers or are mailed in large batches to the payee. Payees that receive this special processing are generally referred to as standard payees. In a "pay-some" system, all the approved payees can be considered to be standard payees. When a user pays a standard payee, there might be different processing lead-times used to calculate the payment and/or processing date of a payment.

When a payment system includes a standard payee list, it might be desirable to present the list to the user, who can then select payees he or she wants to pay. Unfortunately, it is cumbersome to provide this functionality in the client software due to the potential size of this list, which makes it problematic to keep updated and to present to the user. While the list can contain thousands of payee entries, a user will typically need less than ten or twenty entries from the list. It can also be difficult for a user to choose the correct payee entry when the list contains a number of similarly named payees.

Therefore, Open Financial Exchange does not provide a mechanism for delivering these lists to the client. However, there are several ways that an external presentation of such a list can be integrated into the client or server. For example, a payment provider's Web site could present a search engine that assists the user to locate the correct payee. Once identified, the payees can either be imported into the client, or inserted into the user's payee list on the server. In the latter case, synchronizing the payee list will make the newly added payees visible to the client.

12.2.4 Summary – Identifying Payees

Clients can identify a payee in one of four ways:

- Name and address, by means of <PAYEE>, should be identified only once for each payee. Thereafter, clients must use the assigned <PAYEELSTID> and <PAYEEID>. If the clients sends both <PAYEE> and <PAYEELSTID>, it is an implicit payee modification request. If a client sends just <PAYEE>, it is an implicit payee add request. Duplicate payee list entries can occur if clients are not careful to send the payee list ID in subsequent requests.
- Destination bank account <BANKACCTTO> should be done only once for each payee. Thereafter, clients should use the assigned <PAYEELSTID> or <PAYEEID>, as with name and address payees. The <PAYEE> aggregate is required to provide name and address information as a backup to account transfers.
- Payee list ID <PAYEELSTID> after a payee has been added to the list. For payees that have not been assigned a standard payee ID <PAYEEID>, the <PAYEELSTID> is enough to identify the payee in version 2 of the message set.
- Standard payee ID <PAYEEID> for any payee that has been assigned a standard payee ID. This could happen before a closed system makes any payments, or anytime after the server has notified the client that a payee has a standard payee ID. If a <PAYEELSTID> also exists for the payee, it is required in the request and response, in addition to the <PAYEEID>.

Note! *Servers must always assign <PAYEELSTID>s to payees. Once <PAYEELSTID>s have been assigned, clients must always send the <PAYEELSTID>, even if a Payee has both a <PAYEEID> and a <PAYEELSTID>.*

12.3 Identifiers Used in Payment Transactions

Payment transactions use four types of identifiers. It is important to understand the purpose, scope, and life span of these identifiers.

The client-to-request messages assign the Transaction Universal Identifier <TRNUID>. Its purpose is to allow the client to easily match responses from the server to their corresponding requests. A given transaction ID is used only for a client request and the corresponding server response.

The Server Identifier, <SRVRTID> or <RECSRVRTID>, is assigned by the server to a payment "object," which can either be a payment or a recurring payment model (in which case it is named <RECSRVRTID>). Both the client and server use the ID thereafter to refer to the payment or model in any transactions that operate on them. For example, the <SRVRTID> is used to identify a payment in a request to modify or cancel it. The <SRVRTID> is valid for the life span of the payment within the payment system. Similarly the <RECSRVRTID> is valid as long as the associated model exists, that is until the model generates all payments, or the model is canceled. Once a server processes a payment or a model generates all its required payments, the associated <SRVRTID> (or <RECSRVRTID>) is no longer known to the server. Note that the payment system might continue to maintain knowledge of a payment <SRVRTID> or model <RECSRVRTID> for some specified period after it completes processing. This allows clients to access the "completed" status of these operations.

A payment system can assign the Payee Identifier <PAYEEID> to a payee. There is no requirement that all or any payees are assigned a <PAYEEID>. The usage of this identifier will vary by payment system. For example, in "pay-some" systems usually every payee has a payee ID with a scope that is known globally, while in "pay-any" systems there might only be <PAYEEID>s assigned to standard payees. When a payee has an assigned <PAYEEID>, the life span of the ID will depend on its scope. If the scope is global, such as for payees in some "pay-some" systems or those with standard payees, then the <PAYEEID> is expected to be valid as long as that payee is identifiable by ID. If the payee ID is user-specific in scope, then the <PAYEEID> is valid as long as the payee appears in the user's server-hosted payee list.

The Payee List Identifier <PAYEELSTID> is assigned by the server to each entry in a user's server-hosted payee list. The need for this identifier is to support the variety of payee models employed in various payment systems. As discussed above, some payment systems assign a payee identifier <PAYEEID> to every payee (this is particularly the case with pay-some systems); others assign <PAYEEID>s only to standard payees. There are also systems that cannot map a payee billing address to a <PAYEEID> in real time. Also, there are systems that can convert a payee from a standard payee with an assigned <PAYEEID> to one that is identified only by billing address. Therefore, systems employ the <PAYEELSTID> to insure that, in systems where payees will not always have a <PAYEEID>, there is another identifier that can be used to reference every payee. This insures that a client can correctly link payments to their payees. The <PAYEELSTID> must be valid as long as the user's payee list includes the payee it identifies, even if the server subsequently assigns a <PAYEEID> to the payee.

12.4 The Payment Life Cycle

12.4.1 Payment Creation

The client formulates a <PMTRQ> that includes the payee, the date, the amount of the payment, the funding account, and the <PAYACCT>. If supported by the user's payment system, the billing address can specify the payee.

The server will look up the payee in the user's payee list. If it is not already in the table, the server will add it and issue a payee list identifier <PAYEELSTID>. This form of payment request performs an implicit Payee Request <PAYEERQ>, which is equivalent to explicitly adding the payee (by means of a <PAYEERQ>), prior to issuing the <PMTRQ>. It has the advantage of being atomic. If the payment request fails, the payee is not added to the user's payee list. Conversely the payment request will fail if the payee information is invalid.

The server responds to the <PMTRQ> with a Payment Response <PMTRS>. Some servers will not be able to immediately return a payee ID at this point, or might not issue payee IDs for all payees. Therefore the <PAYEELSTID> contained in the response functions as the linkage between the payee and the payment. Payment systems use the <SRVRTID> returned in the <PMTRS> to identify the payment for the length of its instantiation on the payment system.

> ***Note!*** *Servers should generate explicit responses to implicit requests. In other words, implicit payee additions or modifications resulting from a new or changed payment should generate explicit payee add or payee change responses from the server. Such explicit responses are only returned to the client in a SYNC response. If the payment transactions containing implicit payee additions or modifications fail, then the payee actions are not executed, since such a compound payment transaction represents a single unit of work (comprised of both payee and payment actions).*

12.4.2 Payment Modification

Between the time the client schedules a payment and the time the server processes the payment, the client can request changes to the parameters of that payment. For example, the amount or date of the payment can be modified. The system uses the Payment Modification Request <PMTMODRQ> for this purpose, where the <SRVRTID> from the <PMTRS> identifies the targeted payment. The user request must specify the full contents of the payment request, including both modified and unmodified data. However, the <PAYEELSTID> is enough to identify the payee in the <PMTMODRQ> with version 2 of the message set.

Full-featured servers will use <PMTMODRS> messages, conveyed to the client during synchronization <PMTSYNCRS>, to inform the client about changes in the state of the client that occur due to server processing. This would include reporting the date the server actually processed a payment, or it failed due to insufficient funds. Servers that are unable to generate <PMTMODRS> responses for this purpose must support the <PMTINQRQ> message described below.

12.4.3 Payment Status Inquiry

As a scheduled payment progresses through its "life-cycle" on the server, the processing status changes accordingly from "scheduled to be processed" to "was processed" or "failed processing." A processing date is associated with these states. The preferred method for providing updated processing status to a client is by use of server-generated Payment Modification messages <PMTMODRS>, as discussed above. However it is possible that less full-featured servers might have difficulty in implementing this form of notification. In this case, Open Financial Exchange requires such servers to implement the Payment Status Inquiry message <PMTINQRQ>, which provides an interface for the client to explicitly request the processing status of individual payments.

12.4.4 Payment Cancellation

In the interval between successful processing of a <PMTRQ> and the actual processing of the payment, the client can cancel the payment by issuing a Payment Cancellation Request <PMTCANCRQ>. The <SRVRTID> value returned in <PMTRS> identifies the payment.

When a payment system cancels a payment, servers can generate a <PMTCANCRS>. This might occur if the user requests payment cancellation by way of a telephone call to customer support or through an e-mail message. The client will receive this response when performing a payment synchronization <PMTSYNCRQ>/<PMTSYNCRS>.

12.4.5 Delayed Payee Matching

Payment systems that allow payment by payee billing address often perform a matching operation to determine if the payee is a standard payee. If this matching occurs in the processing of a <PMTRQ>, and the server recognizes the payee as a standard one, then the server returns the payee ID and payment parameters in the <EXTDPAYEE> aggregate of the <PMTRS>. However some payment systems will not be able to perform "payee matching" at this point in processing. In this case, the server sends updated payee information to the client by using <PAYEESYNRQ> to synchronize the payee list. The client can link payee information in the <PAYEETRNRS> messages to payments with matching <PAYEELSTID> identifiers.

12.5 Common Payments Aggregates

This section documents several aggregates used throughout the Payments portion of the Open Financial Exchange specification.

12.5.1 Payments Account Information <BPACCTINFO>

Open Financial Exchange uses the payments account information aggregate to download account information from an FI. It includes account number specification in <BANKACCTFROM> as well as the status of the service. In Open Financial Exchange, Banking and Payments share the <BANKACCTFROM> aggregate to identify a specific account. For more information, see section 11.3.1.

Tag	*Description*
<BPACCTINFO>	Payments-account-information aggregate
<BANKACCTFROM>	Bank-account-from aggregate
</BANKACCTFROM>	

Tag	Description
<SVCSTATUS>	Status of the account AVAIL = Available, but not yet requested PEND = Requested, but not yet available ACTIVE = In use
</BPACCTINFO>	

12.5.2 Payment Information <PMTINFO>, <PMTINFO2>

The Payment Information aggregate is used to specify detailed payment information. It is used for both single payments and recurring payments. Clients must send the <PAYEELSTID> and <PAYEEID> if known. Clients send a <PAYEE> aggregate if this is an implicit payee add or modify. See section 12.2.4 on identifying payees, above. The <EXTDPMT> aggregate (see section 12.5.2.2) allows the inclusion of disbursement instructions to be printed with the payment. This aggregate is optional.

Tag	Version	Description
<PMTINFO>	V1 only	
<BANKACCTFROM>		Account-from aggregate, see section 11.3.1
</BANKACCTFROM>		
<TRNAMT>		Payment amount, ***amount*** This amount should be specified as a positive number
Specify payee; either <PAYEEID> or <PAYEE>.		in version 2, PAYEE may be omitted
<PAYEEID>		Server payee identifier (required if assigned). Either <PAYEEID> or <PAYEE> can be sent, but not both. ***SRVRTID***
<PAYEE>		Complete payee billing information, see section 12.5.2.1. Either <PAYEEID> or <PAYEE> can be sent, but not both.
</PAYEE>		
<PAYEELSTID>		Payee list ID (required if assigned), ***SRVRTID***
<BANKACCTTO>		Destination account information, for systems that pay by transfers (<PAYEE> also required)
</BANKACCTTO>		
<EXTDPMT>		Extended Payment aggregate, see section 12.5.2.2, optional
</EXTDPMT>		
<PAYACCT>		User account number with the payee, ***A-32***
<DTDUE>		Payment due date or the date by which payment must be received by payee, ***datetime***
<MEMO>		Memo from user to payee, ***memo***
<BILLREFINFO>		Biller-supplied reference information when paying a bill, if available, ***A-80*** ***Note: If the client user interface has a single field that can contain either free-form memo text or a structured reference number, then the contents of that field should be passed in the <MEMO> tag rather than the <BILLREFINFO> tag.*** Use of this field by country: COUNTRY Interpretation BEL Not present CAN Not present

Tag	Version	Description
		CHE — ***Referenz-Nr***, if applicable DEU — Not present ESP — Not present. FRA — Not present GBR — Not present ITA — ***Coordinate Azienda***, if applicable NLD — ***Betalingskenmerk***, if applicable USA — used to link payment to bill presentment
</PMTINFO>		

Tag	Version	Description
<PMTINFO2>	V2 only	
<PMTTYPE**>**		Country-specific payment type, for those countries in which multiple types of payments are supported by OFX. Use of this field by country: COUNTRY — Possible PMTTYPE values BEL — Not present CAN — Not present CHE — ***ESR, ES, Bankzahlung*** DEU — Not present ESP — Not present. FRA — Not present GBR — Not present ITA — ***Bonifico, RiBa, MAV*** NLD — ***Overschrijving***, ***AcceptGiro*** USA — Not present
<BANKACCTFROM> **</BANKACCTFROM>**		Account-from aggregate, see section 11.3.1
<TRNAMT>		Payment amount, ***amount*** This amount should be specified as a positive number
Specify payee; either <PAYEEID2> or <PAYEE2>.		in version 2, PAYEE may be omitted
<PAYEEID2>		Server payee identifier (required if assigned). Either <PAYEEID2> or <PAYEE2> can be sent, but not both. ***SRVRTID2***
<PAYEE2> **</PAYEE2>**	V2 change	Complete payee billing information, see section 12.5.2.1. Either <PAYEEID2> or <PAYEE2> can be sent, but not both. In version 2 of the message set, if the payee is not a standard payee, then the <PAYEE2> aggregate may be omitted and just the <PAYEELSTID2> supplied to identify the payee.
<PAYEELSTID2>		Payee list ID (required if assigned), ***SRVRTID2***
<BANKACCTTO>		Destination account information, for systems that pay by transfers (<PAYEE> also required)

Tag	*Version*	*Description*
</BANKACCTTO>		
<EXTDPMT>		Extended Payment aggregate, see section 12.5.2.2, optional
</EXTDPMT>		
<PAYACCT>	V2 change	User account number with the payee, ***A-32*** Required in message set version 1. In message set version 2, required if <COUNTRY> = USA or CAN or if <COUNTRY> is not present in signon. Optional otherwise.
<DTDUE>	V2 change	Payment due date or the date by which payment must be received by payee, ***datetime*** In message set version 2, if <DAYSTOPAY> or <XFERDFLTDAYSTOPAY> is –1, this field is interpreted as an execution date (the date that the payer's bank will begin processing the payment) rather than a due date.
<DTAVAIL>		Beneficiary value date. When the funds must be available in <BANKACCTTO>, ***datetime*** Note that for some countries (notably Italy), <DTAVAIL> can be a date in the past. This field is only supported by the server if the <SUPPORTDTAVAIL> tag of the Bill Pay Message Set profile is true (see section 12.11.2). This tag is not used if <COUNTRY> = USA
<ITA.CAUSALE>		***Causale*** code. Reason for the payment, for <COUNTRY> = ITA only, ***N-2***
<PMTFOR>		Name of the person on whose behalf the payment is being made. If not present, payment is assumed to be on behalf of the owner of <BANKACCTFROM>, ***A-32*** This tag is currently used only for <COUNTRY> = ITA.
<BOOKINGTEXT>		Text with which this payment should be identified in the payer's bank statement, ***A-40*** This tag is currently used only for <COUNTRY> = CHE.
<MEMO2>		Memo from user to payee, ***memo2***
<BILLREFINFO>		Biller-supplied reference information when paying a bill, if available, ***A-80*** ***Note: If the client user interface has a single field that can contain either free-form memo text or a structured reference number, then the contents of that field should be passed in the <MEMO> tag rather than the <BILLREFINFO> tag.*** Use of this field by country: COUNTRY — Interpretation BEL — Not present CAN — Not present CHE — ***Referenz-Nr***, if applicable DEU — Not present ESP — Not present. FRA — Not present GBR — Not present ITA — ***Coordinate Azienda***, if applicable NLD — ***Betalingskenmerk***, if applicable USA — used to link payment to bill presentment
</PMTINFO2>		

12.5.2.1 Payee <PAYEE>, <PAYEE2>

<PAYEE> specifies a complete billing address for a payee.

Tag	*Version*	*Description*
<PAYEE>	V1 only	
<NAME>		Name of payee. ***A-32***
<ADDR1>		Payee's address line 1, ***A-32***
<ADDR2>		Payee's address line 2, ***A-32***
<ADDR3>		Payee's address line 3, ***A-32***
<CITY>		Payee's city, ***A-32***
<STATE>		Payee's state, ***A-5***
<POSTALCODE>		Payee's zip code, ***A-11***
<COUNTRY>		Payee's country; 3-letter country code from ISO/DIS-3166, ***A-3***
<PHONE>		Payee's telephone number, ***A-32***
</PAYEE>		

Tag	*Version*	*Description*
<PAYEE2>	V2 only	
<NAME>		Name of payee. ***A-32***
<ADDR1>		Payee's address line 1, ***A-32***
		Required if <COUNTRY> = USA or CAN or if <COUNTRY> is not present, optional otherwise
<ADDR2>		Payee's address line 2, ***A-32***
<ADDR3>		Payee's address line 3, ***A-32***
<CITY>		Payee's city, ***A-32***
		Required if <COUNTRY> = USA or CAN or if <COUNTRY> is not present, optional otherwise
<STATE>		Payee's state, ***A-5***
		Required if <COUNTRY> = USA or CAN or if <COUNTRY> is not present, optional otherwise
<POSTALCODE>		Payee's zip code, ***A-11***
		Required if <COUNTRY> = USA or CAN or if <COUNTRY> is not present, optional otherwise
<COUNTRY>		Payee's country; 3-letter country code from ISO/DIS-3166, ***A-3***
<PHONE>		Payee's telephone number, ***A-32***
		Required if <COUNTRY> = USA or CAN or if <COUNTRY> is not present, optional otherwise
</PAYEE2>		

12.5.2.2 Extended Payment <EXTDPMT>

The Extended Payment aggregate provides the payee with information for applying a payment across multiple invoices. It is structured to allow for electronic processing of the invoice date, and allows multiple invoices, as well as multiple line items per invoice, to be specified.

In this case, <EXTDPMT> can specify a block of free text to be transmitted with the payment, by using the <EXTDPMTDSC> instead of the <EXTDPMTINV> element.

The amount fields for <ADJAMT> and <LITMAMT> can be negative

The <INVPAIDAMT> should be supplied when <INVOICE> is used inside an <EXTDPMT> as shown. It is marked optional in accordance with the use of <INVOICE> in section 14.4.2.2.2.

Tag	*Version*	*Description*
<EXTDPMT>		Extended Payment aggregate
<EXTDPMTFOR>		INDIVIDUAL or BUSINESS. Indicates whether the payment is for an individual or business account. This allows the payment processor to remit payments to the appropriate address for consumers or businesses.
<EXTDPMTCHK>	V1 only	Check number to use for this payment. Overrides "next check in range." ***N-10***
<EXTDPMTCHK2>	V2 only	Check number to use for this payment. Overrides "next check in range." ***A-12***
Payment description. Either or both of the following aggregates: <EXTDPMTDSC> or <EXTDPMTINV>.		
<EXTDPMTDSC>		Free text to communicate with the payment, ***A-255***
<EXTDPMTINV>		Invoices aggregate
<INVOICE>		Start tag for the invoice aggregate. There can be one or more invoices per payment request.
<INVNO>		Invoice number associated with the payment. ***A-32***
<INVTOTALAMT>		This value represents the total invoice amount, ***amount***
		This amount should be specified as a positive number
<INVPAIDAMT>		This value represents the amount of the invoice being paid, ***amount***
		Always required in bill payment
		This amount should be specified as a positive number
<INVDATE>		Date to apply the invoice, ***datetime***
<INVDESC>		Invoice description, ***A-80***
<DISCOUNT>	V1 only	Discount aggregate; only one discount aggregate per invoice
<DSCRATE>		Discount rate, optional if <DSCAMT> tag is used, ***rate***
		Either <DSCRATE> or <DSCAMT>, but not both.
<DSCAMT>		Discount amount, optional if <DSCRATE> tag is used, ***amount***
		This amount should be specified as a positive number
		Either <DSCRATE> or <DSCAMT>, but not both.
<DSCDATE>		Date to apply the discount, ***datetime***
<DSCDESC>		Discount description, ***A-80***
</DISCOUNT>		
<DISCOUNT2>	V2 only	Discount aggregate; only one discount aggregate per invoice

Tag	Version	Description
Specify discount, either DSCRATE or DSCAMT but not both (as of V2)		
<DSCRATE>	V2 change	Discount rate, optional if <DSCAMT> tag is used, ***rate***
		Either <DSCRATE> or <DSCAMT>, but not both.
<DSCAMT>	V2 change	Discount amount, optional if <DSCRATE> tag is used, ***amount***
		This amount should be specified as a positive number
		Either <DSCRATE> or <DSCAMT>, but not both.
<DSCDATE>		Date to apply the discount, ***datetime***
<DSCDESC>		Discount description, ***A-80***
</DISCOUNT2>		
<ADJUSTMENT>		Adjustment aggregate; only one adjustment aggregate per invoice
<ADJNO>		Adjustment number associated with the payment, ***A-32***
<ADJDESC>		Adjustment description, ***A-80***
<ADJAMT>		Amount of the adjustment, ***amount***
		This amount should be signed + or -, as appropriate. A positive adjustment means that the payment amount has been reduced.
<ADJDATE>		Date of adjustment, ***datetime***
</ADJUSTMENT>		
<LINEITEM>		Line item aggregate; there can be multiple line items per invoice
<LITMCODE>	V2 only	Posting Code ***A-32***
<LITMAMT>		Amount of the line item, ***amount***
		This amount should be signed + or -, as appropriate. A positive line item amount is an addition to the payment amount, and a negative line item is a discount or reduction in the payment amount.
<LITMDESC>		Line item description, ***A-80***
</LINEITEM>		
</INVOICE>		
</EXTDPMTINV>		
</EXTDPMT>		

12.5.2.3 Extended Payee <EXTDPAYEE>

The Extended Payee aggregate communicates a payee identifier to the client. It also contains the processing day parameters for a payee. It can be sent to the client for any payee whose processing day parameters are different from the processor's default values, even for payees with no <PAYEEID>.

Tag	Version	Description
<EXTDPAYEE>		Extended-payee aggregate
<PAYEEID>	V1 only	Server-assigned payee ID, ***SRVRTID***
<PAYEEID2>	V2 only	Server-assigned payee ID, ***SRVRTID2***
<IDSCOPE>		Scope of the payee ID; one of (GLOBAL, USER), where GLOBAL = payee ID valid across the entire payment system USER = payee ID valid with all FI accounts set up for the user's payments account

Tag	Version	Description
<NAME>		Standard payee name, ***A-32***
<DAYSTOPAY>	V2 change	Minimum number of business days needed to process, ***N-3***
		In version 2 of the message set, If the value of <DAYSTOPAY> is -1, then <DTDUE> in <PMTINFO> is interpreted as an execution date (the date that the payer's bank will begin processing the payment) rather than a due date.
</EXTDPAYEE>		

12.5.2.4 Payment Processing Status <PMTPRCSTS>

The Payment Processing Status aggregate contains the current processing status for a payment. The interpretation of the date value depends on the value of <PMTPRCCODE>.

Tag	Description
<PMTPRCSTS>	
<PMTPRCCODE>	See table 12.6.2.1
<DTPMTPRC>	Payment processing date; interpretation depends on <PMTPRCCODE>, ***datetime***
</PMTPRCSTS>	Ending tag for payment processing status

12.6 Payments Functions

Payments functions allow a client to create a Payment Request to pay a bill on a specified date. The Payment Request identifies the payee and the amount to pay. Because the flow of money is unambiguous, bill payment amounts are usually specified as positive numbers. See tables for details.

Client Sends	Server Responds
Account information	
Payment date	
Amount	
Payee address, list ID, transfer acct, or standard ID	
	Payment status
	Check number
	Server-assigned ID

From the time the client issues a Payment Request until it is paid, the client can modify the transaction through the Payment Modification Request, <PMTMODRQ>; see section 12.4.2. This request allows payment parameters such as the payment date and payment amount to be changed.

Client Sends	Server Responds
Account information	
Server-assigned ID	
Information to change:	
Payment date,	
Amount,...	
	Acknowledgment or Error

The client can cancel a Payment Request with a Payment Cancellation Request, <PMTCANCRQ>; see section 12.4.4.

Client Sends	*Server Responds*
Account information Server-assigned ID	
	Acknowledgment or Error

12.6.1 Payment Creation

A Payment Request is used to schedule an electronic payment. The server responds with a Payment Response. Separate transactions are provided for modifying and canceling a Payment Request.

12.6.1.1 Payment Request <PMTRQ>

The <PMTRQ> request must appear within a <PMTTRNRQ> transaction wrapper.

Tag	*Version*	*Description*
<PMTRQ>		Payment-request aggregate
<PMTINFO>	V1 only	Payment Information aggregate, see section 12.5.2
</PMTINFO>		
<PMTINFO2>	V2 only	Payment Information aggregate, see section 12.5.2
</PMTINFO2>		
</PMTRQ>		

12.6.1.2 Payment Response <PMTRS>

The server sends a Payment Response in response to a Payment Request. The processing status code for a new payment is normally WILLPROCESSON, but in the case of synchronization it can return other status codes.

> ***Note:*** *If the <PMTRQ> created a new payee, the server must create and store a payee response that would be available for a payee synchronization request. The server need only send the <PMTRS> as a response to the <PMTRQ>, but the ability to send the payee synchronization response is required in case another client logs on and requests only a payee synchronization.*

The <PMTRS> response must appear within a <PMTTRNRS> transaction wrapper.

Tag	*Version*	*Description*
<PMTRS>		Payment-response aggregate
<SRVRTID>	V1 only	ID assigned by the server to the payment being created, ***SRVRTID***
<SRVRTID2>	V2 only	ID assigned by the server to the payment being created, ***SRVRTID2***
<PAYEELSTID>	V1 only	Server-assigned payee list record ID for this payee, ***SRVRTID***
<PAYEELSTID>	V2 only	Server-assigned payee list record ID for this payee, ***SRVRTID2***
<CURDEF>		Default currency for the Recurring Payment Response, ***currsymbol***
<PMTINFO>	V1 only	Payment Information aggregate, see section 12.5.2
</PMTINFO>		

Tag	Version	Description
<PMTINFO2>	V2 only	Payment Information aggregate, see section 12.5.2
</PMTINFO2>		
<EXTDPAYEE>		Standard payee information if payee is a standard payee, or payee has non-default processing day parameters; see section 12.5.2.3
</EXTDPAYEE>		
<CHECKNUM>		Check number, ***A-12***
<PMTPRCSTS>		Payment processing status
</PMTPRCSTS>		
<RECSRVRTID>	V1 only	References the payment if it was generated by a recurring payment, ***SRVRTID***
<RECSRVRTID2>	V2 only	References the payment if it was generated by a recurring payment, ***SRVRTID2***
</PMTRS>		

12.6.1.3 Status Codes

Code	Meaning
0	Success (INFO)
2000	General error (ERROR)
2002	General account error (ERROR)
2006	Source account not found (ERROR)
2007	Source account closed (ERROR)
2008	Source account not authorized (ERROR)
2009	Destination account not found (ERROR)
2010	Destination account closed (ERROR)
2011	Destination account not authorized (ERROR)
2012	Invalid amount (ERROR)
2014	Date too soon (ERROR)
2015	Date too far in future (ERROR)
2019	Duplicate request (ERROR)
10501	Invalid payee (ERROR)
10502	Invalid payee address (ERROR)
10503	Invalid payee account number (ERROR)
10510	Invalid payee ID (ERROR)
10511	Invalid payee city (ERROR)
10512	Invalid payee state (ERROR)
10513	Invalid payee postal code (ERROR)
10517	Invalid payee name (ERROR)
10519	Invalid payee list ID (ERROR)

12.6.1.4 Discussion

Once the server has assigned a payee identifier <PAYEEID> to the payee, it includes the <EXTDPAYEE> in any <PMTRS> for that transaction. If the <EXTDPAYEE> aggregate is present in the Payment Response <PMTRS>, the client records the standard payee information for use in future payments to the same payee.

When a payment is made using the <PAYEE> aggregate, and no <PAYEELSTID> is present, the payee is implicitly added to the payee list. This is therefore equivalent to first transmitting a <PAYEERQ> for the payee. For payment systems that can immediately return payee IDs, it is preferable to use the single <PMTRQ> message to both add the payee and create the payment. If either operation fails, the server will not complete the other. If <PAYEELSTID> and <PAYEE> are both included, it is equivalent to sending a <PAYEEMODRQ>.

The <PMTRS> response will include the <EXTDPAYEE> aggregate if the processor has assigned a payee ID to the payee specified in the payment. It will also appear in the response when the payee has no assigned ID, but has processing day parameters that different from the processor's defaults for these values. This might occur, for instance, if the processor notes that the zip code of the payee indicates a certain proximity to the payer, and therefore wishes to offer a shorter <DAYSTOPAY> value.

12.6.2 Payment Modification

The Payment Modification Request allows a client to modify a previously scheduled payment. When modifying a payment, the client must specify all of the tags within the <PMTINFO> aggregate that were specified during the payment creation or previous modification, not just the tags that it wants to modify. The ability to modify some tag values might not be supported by all servers.

> ***Note!*** *In version 2 of the message set, the client specifies all of the tags within the <PMTINFO> aggregate that were specified during the payment creation or previous modification as outlined above. The exception to this is <PAYEE>, which can be omitted in version 2 if there is no intent to modify the payee, since the <PAYEELSTID> would identify the payee.*

12.6.2.1 Payment Processing Status Values <PMTPRCCODE>

Value	*Description*
WILLPROCESSON	Will be processed on <DTPMTPRC>
PROCESSEDON	Was processed for payment on <DTPMTPRC>
NOFUNDSON	Funds not available to make payment on <DTPMTPRC>
FAILEDON	Unable to make payment for unspecified reasons on <DTPMTPRC>
CANCELEDON	User canceled payment on <DTPMTPRC>

12.6.2.2 Payment Modification Request <PMTMODRQ>

The client sends a Payment Modification Request to request modification of a payment. The client must provide the full <PMTINFO> including both changed and unchanged values.

The client is free to modify any data in <PMTINFO>, excluding the payee name (the <NAME> element of <PAYEE>), payee ID <PAYEEID>, and <BANKACCTFROM>.

The <PMTMODRQ> request must appear within a <PMTTRNRQ> transaction wrapper.

Tag	*Version*	*Description*
<PMTMODRQ>		Modification-request this references
<SRVRTID>	V1 only	ID assigned by the server to the payment being modified, ***SRVRTID***
<SRVRTID2>	V2 only	ID assigned by the server to the payment being modified, ***SRVRTID2***
<PMTINFO>	V1 only	Payment Information aggregate, see section 12.5.2
</PMTINFO>		

Tag	Version	Description
<PMTINFO2>	V2 only	Payment Information aggregate, see section 12.5.2
</PMTINFO2>		
</PMTMODRQ>		

12.6.2.3 Payment Modification Response <PMTMODRS>

The server sends a Payment Modification Response in response to a Payment Modification Request.

> ***Note:*** *If the <PMTMODRQ> created a new payee, the server must create and store a payee response that would be available for a payee synchronization request. The server need only send the <PMTMODRS> as a response to the <PMTMODRQ>, but the ability to send the payee synchronization response is required in case another client logs on and requests only a payee synchronization.*

The <PMTMODRS> response must appear within a <PMTTRNRS> transaction wrapper.

Tag	Version	Description
<PMTMODRS>		Payment-modification-response this references
<SRVRTID>	V1 only	ID assigned by the server to the payment being modified, ***SRVRTID***
<SRVRTID2>	V2 only	ID assigned by the server to the payment being modified, ***SRVRTID2***
<PMTINFO>	V1 only	Payment Information aggregate, see section 12.5.2
</PMTINFO>		
<PMTINFO2>	V2 only	Payment Information aggregate, see section 12.5.2
</PMTINFO2>		
<PMTPRCSTS>		Payment processing status, see section 12.5.2.4
</PMTPRCSTS>		
</PMTMODRS>		

12.6.2.4 Status Codes

Code	Meaning
0	Success
2000	General error (ERROR)
2002	General account error (ERROR)
2006	Source account not found (ERROR)
2007	Source account closed (ERROR)
2008	Source account not authorized (ERROR)
2009	Destination account not found (ERROR)
2010	Destination account closed (ERROR)
2011	Destination account not authorized (ERROR)
2012	Invalid amount (ERROR)
2014	Date too soon (ERROR)
2015	Date too far in future (ERROR)
2016	Already committed (ERROR)
2017	Already canceled (ERROR)
2018	Unknown server ID (ERROR)

Code	Meaning
2019	Duplicate request (ERROR)
10501	Invalid payee (ERROR)
10502	Invalid payee address (ERROR)
10503	Invalid payee account number (ERROR)
10505	Cannot modify element (ERROR)
10510	Invalid payee ID (ERROR)
10511	Invalid payee city (ERROR)
10512	Invalid payee state (ERROR)
10513	Invalid payee postal code (ERROR)
10517	Invalid payee name (ERROR)
10519	Invalid payee list ID (ERROR)

12.6.2.5 Discussion

Servers can initiate <PMTMODRS> messages to communicate changes in the processing status of a payment as it moves through the payment system. This mechanism allows a client to capture the updated status of payments every time it synchronizes.

Implicit payee changes contained in a payment modification transaction do not affect any other existing pending payments. The changes are propagated to the server's payee list and affect payments to that payee as subsequently initiated by the client after the change, or as subsequently spawned from a recurring model. Explicit payee changes are not propagated to payments pending for the changed payee at the time of the change.

12.6.3 Payment Cancellation

The Payment Cancellation Request allows a client to cancel a previously scheduled payment created with a Payment Request (<PMTRQ> in section 12.6.1.1).

12.6.3.1 Request <PMTCANCRQ>

The client sends a Payment Cancellation to cancel a scheduled payment request.

The <PMTCANCRQ> request must appear within a <PMTTRNRQ> transaction wrapper.

Tag	Version	Description
<PMTCANCRQ>		Cancellation-request this references
<SRVRTID>	V1 only	ID assigned by the server to the payment being canceled, ***SRVRTID***
<SRVRTID2>	V2 only	ID assigned by the server to the payment being canceled, ***SRVRTID2***
</PMTCANCRQ>		

12.6.3.2 Response <PMTCANCRS>

The server sends a Payment Cancellation Response in response to a Payment Cancellation Request.

The <PMTCANRS> response must appear within a <PMTTRNRS> transaction wrapper.

Tag	Version	Description
<PMTCANCRS>		Cancellation-response this references
<SRVRTID>	V1 only	ID assigned by the server to the payment being canceled, ***SRVRTID***

Tag	Version	Description
<SRVRTID2>	V2 only	ID assigned by the server to the payment being canceled, ***SRVRTID2***
</PMTCANCRS>		

12.6.3.3 Status Codes

Code	Meaning
0	Success
2000	General error (ERROR)
2016	Already committed (ERROR)
2017	Already canceled (ERROR)
2018	Unknown server ID (ERROR)
2019	Duplicate request (ERROR)

12.6.4 Payment Status Inquiry

The Payment Status Inquiry Request allows a client to obtain the current processing status of a payment from the server.

12.6.4.1 Request <PMTINQRQ>

The client sends a Payment Status Inquiry Request to obtain the current processing status of a payment.

The <PMTINQRQ> request must appear within a <PMTINQTRNRQ> transaction wrapper.

Tag	Version	Description
<PMTINQRQ>		Payment-status-inquiry-request aggregate
<SRVRTID>	V1 only	ID assigned by the server to the payment being queried, ***SRVRTID***
<SRVRTID2>	V2 only	ID assigned by the server to the payment being queried, ***SRVRTID2***
</PMTINQRQ>		

12.6.4.2 Response <PMTINQRS>

The server sends a Payment Status Inquiry Response in response to a Payment Status Inquiry Request.

The <PMTINQRS> response must appear within a <PMTINQTRNRS> transaction wrapper.

Tag	Version	Description
<PMTINQRS>		Payment-status-inquiry-response aggregate
<SRVRTID>	V1 only	ID assigned by the server to the payment being queried, ***SRVRTID***
<SRVRTID2>	V2 only	ID assigned by the server to the payment being queried, ***SRVRTID2***
<PMTPRCSTS>		Payment processing status
</PMTPRCSTS>		
<CHECKNUM>		Check number assigned by the server to this payment, ***A-12***
</PMTINQRS>		

12.6.4.3 Status Codes

Code	Meaning
0	Success (INFO)

Code	Meaning
2000	General error (ERROR)
2018	Unknown server ID (ERROR)
2019	Duplicate request (ERROR)

12.7 Recurring Payments

Recurring payments are used when a payment is to be made repeatedly at some known interval. Setting up a recurring payment is similar to creating an individual payment, but with additional information about the frequency and number of payments. After a recurring payment is created, the server will generate payments transactions when there are a specified number of days remaining until the next projected payment is due (usually 30 days). The client will be made aware of any generated payment transactions through the synchronization process. Chapters 10 and 11, "Recurring Transactions" and "Banking," provide additional details on models and recurring transactions, and define the recurring transaction aggregates.

The table below lists the functional elements for creating a recurring payment:

Client Sends	Server Responds
Account information	
Payment frequency	
Number of payments	
Payment date	
Amount	
Payee address, list ID or payee ID	
	Standard payee information
	Server-assigned ID

The table below lists the functional elements for modifying a recurring payment:

Client Sends	Server Responds
Account information	
Server-assigned ID	
Information to change:	
Payment frequency,	
Number of payments,	
Payment date,	
Amount,...	
	Acknowledgment or Error

The table below lists the functional elements for canceling a recurring payment:

Client Sends	Server Responds
Account information	
Server-assigned ID	
	Acknowledgment or Error

12.7.1 Creating a Recurring Payment

Use a Recurring Payment Request to set up a recurring electronic payment. The user can specify the frequency and duration of the payments using the Recurring Instructions aggregate <RECURRINST>. The <PMTINFO> aggregate (see section 12.5.2) specifies the payment information for the model, as well as the initial and final amounts (if present and where applicable)

12.7.1.1 Request <RECPMTRQ>

The <RECPMTRQ> request must appear within a <RECPMTTRNRQ> transaction wrapper.

Tag	*Version*	*Description*
<RECPMTRQ>		Recurring-payment-request aggregate
<RECURRINST>		Recurring Instructions aggregate, see section 10.2.
</RECURRINST>		
<PMTINFO>	V1 only	Payment-information aggregate, see section 12.5.2
</PMTINFO>		
<PMTINFO2>	V2 only	Payment-information aggregate, see section 12.5.2
</PMTINFO2>		
<INITIALAMT>		Amount of the initial payment, if different than the following payments, ***amount***
		This amount should be specified as a positive number
<FINALAMT>		Amount of the final payment, if different than the preceding payments, ***amount***
		This amount should be specified as a positive number
</RECPMTRQ>		

12.7.1.2 Response <RECPMTRS>

The server sends a Recurring Payment Response upon receipt of a Recurring Payment Request.

The <RECPMTRS> response must appear within a <RECPMTTRNRS> transaction wrapper.

Tag	*Version*	*Description*
<RECPMTRS>		Recurring-payment-response aggregate
<RECSRVRTID>	V1 only	Server-assigned ID for this transaction, ***SRVRTID***
<RECSRVRTID2>	V2 only	Server-assigned ID for this transaction, ***SRVRTID2***
<PAYEELSTID>	V1 only	Server-assigned record ID for this payee record, ***SRVRTID***
<PAYEELSTID2>	V2 only	Server-assigned record ID for this payee record, ***SRVRTID2***
<CURDEF>		Default currency for the Recurring Payment Response, ***currsymbol***
<RECURRINST>		Recurring-instructions aggregate, see section 10.2.
</RECURRINST>		
<PMTINFO>	V1 only	Payment-information aggregate, see section 12.5.2
</PMTINFO>		
<PMTINFO2>	V2 only	Payment-information aggregate, see section 12.5.2
</PMTINFO2>		

Tag	Version	Description
<INITIALAMT>		Amount of the initial payment, if different than the following payments, ***amount*** This amount should be specified as a positive number
<FINALAMT>		Amount of the final payment, if different than the preceding payments, ***amount*** This amount should be specified as a positive number
<EXTDPAYEE> </EXTDPAYEE>		Extended payee information, see section 12.5.2.3.
</RECPMTRS>		

12.7.1.3 Status Codes

Code	Meaning
0	Success (INFO)
2000	General error (ERROR)
2002	General account error (ERROR)
2006	Source account not found (ERROR)
2007	Source account closed (ERROR)
2008	Source account not authorized (ERROR)
2009	Destination account not found (ERROR)
2010	Destination account closed (ERROR)
2011	Destination account not authorized (ERROR)
2012	Invalid amount (ERROR)
2014	Date too soon (ERROR)
2015	Date too far in future (ERROR)
2019	Duplicate request (ERROR)
10501	Invalid payee (ERROR)
10502	Invalid payee address (ERROR)
10503	Invalid payee account number (ERROR)
10508	Invalid frequency (ERROR)
10510	Invalid payee ID (ERROR)
10511	Invalid payee city (ERROR)
10512	Invalid payee state (ERROR)
10513	Invalid payee postal code (ERROR)
10517	Invalid payee name (ERROR)
10519	Invalid payee list ID (ERROR)

12.7.1.4 Discussion

The <DTDUE> element of <PMTINFO> specifies payment due date or the date by which the first payment must be received by payee (see section 12.5.2).

12.7.2 Recurring Payment Modification

The client sends a Recurring Payment Modification Request to request modifications to a recurring payment previously created with a Recurring Payment Request. The payment frequency <RECURRINST>, the payment parameters <PMTINFO>, or both, can be changed.

12.7.2.1 Request <RECPMTMODRQ>

The client sends a Recurring Payment Modification Request to request changes to a recurring payment model.

As with individual payments, the client is free to modify any data in <PMTINFO>, excluding the payee name (the <NAME> element of <PAYEE>), payee ID <PAYEEID>, and <BANKACCTFROM>. Clients can modify both elements in the <RECURRINST> aggregate (i.e. <NINSTS> and <FREQ>). Client should send the original number of payments scheduled if there is no change. If there is a change in the number of payments scheduled, clients should send the new number of payments.

> ***Note!*** *A <RECPMTMODRQ> that modifies pending payments via the <MODPENDING> flag is a compound transaction, and generates the appropriate explicit payment responses that reflect such modifications, which are returned to the client via synchronization.*

The <RECPMTMODRQ> request must appear within a <RECPMTTRNRQ> transaction wrapper.

Tag	*Version*	*Description*
<RECPMTMODRQ>		Modification-request aggregate
<RECSRVRTID>	V1 only	ID assigned by the server to the payment being modified, ***SRVRTID***
<RECSRVRTID2>	V2 only	ID assigned by the server to the payment being modified, ***SRVRTID2***
<RECURRINST>		Recurring Instructions aggregate, see section 10.2
</RECURRINST>		
<PMTINFO>	V1 only	Payment-Information aggregate, see section 12.5.2
</PMTINFO>		
<PMTINFO2>	V2 only	Payment-Information aggregate, see section 12.5.2
</PMTINFO2>		
<INITIALAMT>		Amount of the initial payment, if different than the following payments, ***amount***
		This amount should be specified as a positive number
<FINALAMT>		Amount of the final payment, if different than the preceding payments, ***amount***
		This amount should be specified as a positive number
<MODPENDING>		Modify pending flag
		If the client sets this flag, the server must modify pending and future payments. ***Boolean***
</RECPMTMODRQ>		

12.7.2.2 Response <RECPMTMODRS>

The server sends a Recurring Payment Modification Response in response to a Recurring Payment Modification Request.

The <RECPMTMODRS> response must appear within a <RECPMTTRNRS> transaction wrapper.

Tag	*Version*	*Description*
<RECPMTMODRS>		Modification-response aggregate
<RECSRVRTID>	V1 only	ID assigned by the server to the payment being modified, ***SRVRTID***
<RECSRVRTID2>	V2 only	ID assigned by the server to the payment being modified, ***SRVRTID2***
<RECURRINST>		Recurring-Instructions aggregate, see section 10.2
</RECURRINST>		
<PMTINFO>	V1 only	Payment-Information aggregate, see section 12.5.2
</PMTINFO>		
<PMTINFO2>	V2 only	Payment-Information aggregate, see section 12.5.2
</PMTINFO2>		
<INITIALAMT>		Amount of the initial payment, if different than the following payments, ***amount***
		This amount should be specified as a positive number
<FINALAMT>		Amount of the final payment, if different than the preceding payments, ***amount***
		This amount should be specified as a positive number
<MODPENDING>		Y if the client requested that the server modify pending and future payments. N if the client did not request that the server modify pending and future payments., ***Boolean***
</RECPMTMODRS>		

12.7.2.3 Status Codes

Code	*Meaning*
0	Success (INFO)
2000	General error (ERROR)
2002	General account error (ERROR)
2006	Source account not found (ERROR)
2007	Source account closed (ERROR)
2008	Source account not authorized (ERROR)
2009	Destination account not found (ERROR)
2010	Destination account closed (ERROR)
2011	Destination account not authorized (ERROR)
2012	Invalid amount (ERROR)
2014	Date too soon (ERROR)
2015	Date too far in future (ERROR)
2019	Duplicate request (ERROR)

Code	Meaning
10501	Invalid payee (ERROR)
10502	Invalid payee address (ERROR)
10503	Invalid payee account number (ERROR)
10508	Invalid frequency (ERROR)
10511	Invalid payee city (ERROR)
10512	Invalid payee state (ERROR)
10513	Invalid payee postal code (ERROR)
10517	Invalid payee name (ERROR)
10518	Unknown model ID (ERROR)
10519	Invalid payee list ID (ERROR)

12.7.3 Recurring Payment Cancellation

The client sends a Recurring Payment Cancellation Request to cancel a recurring payment previously created with a Recurring Payment Request.

12.7.3.1 Request <RECPMTCANCRQ>

The <RECPMTCANCRQ> request must appear within a <RECPMTTRNRQ> transaction wrapper.

> ***Note!*** *A <RECPMTCANRQ> that cancels pending payments via the <CANPENDING> flag is a compound transaction, and generates the appropriate explicit payment responses that reflect such cancellations, which are returned to the client via synchronization.*

Tag	Version	Description
<RECPMTCANCRQ>		Cancellation-request aggregate
<RECSRVRTID>	V1 only	ID assigned by the server to the payment being canceled, ***SRVRTID***
<RECSRVRTID2>	V2 only	ID assigned by the server to the payment being canceled, ***SRVRTID2***
<CANPENDING>		Cancel pending flag If the client sets this flag, the server must cancel pending and future payments. ***Boolean***
</RECPMTCANCRQ>		

12.7.3.2 Response <RECPMTCANCRS>

The server sends a Recurring Payment Cancellation Response in response to a Recurring Payment Cancellation Request.

The <RECPMTCANCRS> response must appear within a <RECPMTTRNRS> transaction wrapper.

Tag	Version	Description
<RECPMTCANCRS>		Modification-request aggregate
<RECSRVRTID>	V1 only	ID assigned by the server to the payment being modified, ***SRVRTID***
<RECSRVRTID2>	V2 only	ID assigned by the server to the payment being modified, ***SRVRTID2***
<CANPENDING>		Y if the client requested that the server cancel pending and future payments. N if the client did not request that the server cancel pending and future payments. ***Boolean***
</RECPMTCANCRS>		

12.7.3.3 Status Codes

Code	Meaning
0	Success
2000	General error (ERROR)
2019	Duplicate request (ERROR)
10518	Unknown model ID (ERROR)

12.8 Payment Mail

Users can correspond by way of e-mail to resolve problems or ask questions about their payments accounts. This function makes use of the general Open Financial Exchange e-mail facility, which is described in Chapter 9, "Customer to FI Communication."

12.8.1 Payment Mail Request and Response

12.8.1.1 Request <PMTMAILRQ>

The <PMTMAILRQ> allows a client to issue an e-mail to the payments processor. If the message refers to a specific payment, then both <SRVRTID> and <PMTINFO> are required to identify the payment to the processor.

The <PMTMAILRQ> request must appear within a <PMTMAILTRNRQ> transaction wrapper.

Tag	Version	Description
<PMTMAILRQ>		Payment e-mail-request aggregate
<MAIL>		General e-mail aggregate
</MAIL>		
<SRVRTID>	V1 only	Transaction ID of the payment that is the subject of the correspondence, ***SRVRTID***
<SRVRTID2>	V2 only	Transaction ID of the payment that is the subject of the correspondence, ***SRVRTID2***
<PMTINFO>	V1 only.	Payment Information aggregate, see section 12.5.2
</PMTINFO>		
<PMTINFO2>	V2 only.	Payment Information aggregate, see section 12.5.2
</PMTINFO2>		
</PMTMAILRQ>		

12.8.1.2 Response <PMTMAILRS>

The server sends <PMTMAILRS> in response to a Payment E-mail request.

The <PMTMAILRS> response must appear within a <PMTMAILTRNRS> transaction wrapper.

Tag	Version	Description
<PMTMAILRS>		Payment e-mail-response aggregate
<MAIL>		General e-mail aggregate, see Chapter 9, "Customer to FI Communication."
</MAIL>		
<SRVRTID>	V1 only	Transaction ID of the payment that is the subject of the correspondence, ***SRVRTID***
<SRVRTID2>	V2 only	Transaction ID of the payment that is the subject of

Tag	Version	Description
		the correspondence, ***SRVRTID2***
<PMTINFO>	V1 only.	Payment Information aggregate, see section 12.5.2
</PMTINFO>		
<PMTINFO2>	V2 only.	Payment Information aggregate, see section 12.5.2
</PMTINFO2>		
</PMTMAILRS>		

12.8.1.3 Status Codes

Code	Meaning
0	Success (INFO)
2000	General error (ERROR)
2002	General account error (ERROR)
2003	Account not found (ERROR)
2004	Account closed (ERROR)
2005	Account not authorized (ERROR)
2018	Unknown server ID (ERROR)
2019	Duplicate request (ERROR)
16500	HTML not allowed (ERROR)
16501	Unknown mail To: (ERROR)

12.8.2 Payment Mail Synchronization

Payment mail is subject to synchronization. The scope of the synchronization request is all of the accounts for which the user might have sent mail, not a specific account.

12.8.2.1 Request <PMTMAILSYNCRQ>

Tag	Version	Description
<PMTMAILSYNCRQ>		Synchronization-request aggregate
Client synchronization option; <TOKEN>, <TOKEN2>, <TOKENONLY>, or <REFRESH>		
<TOKEN>	V1 only	Previous value of <TOKEN> received for this type of synchronization request from server; 0 for first-time requests; ***token***
<TOKEN2>	V2 only	Previous value of <TOKEN2> received for this type of synchronization request from server; 0 for first-time requests; ***token2***
<TOKENONLY>		Request for just the current <TOKEN> without the history, ***Boolean***
<REFRESH>		Request for refresh of current state, ***Boolean***
<REJECTIFMISSING>		If Y, do not process requests if client <TOKEN> is out of date, ***Boolean***
<INCIMAGES>		Y if the client accepts mail with images in the message body. N if the client does not accept mail with images in the message body. ***Boolean***
<USEHTML>		Y if client wants an HTML response, N if client wants plain text, ***Boolean***
<PMTMAILTRNRQ>		Payment-mail transactions (0 or more)
</PMTMAILTRNRQ>		
</PMTMAILSYNCRQ>		

12.8.2.2 Response <PMTMAILSYNCRS>

Tag	*Version*	*Description*
<PMTMAILSYNCRS>		Synchronization-response aggregate
<TOKEN>	V1 only	New synchronization token, ***token***
<TOKEN2>	V2 only	New synchronization token, ***token2***
<LOSTSYNC>		Y if the token in the synchronization request is older than the earliest entry in the server's history table. In this case, some responses have been lost. N if the token in the synchronization request is newer than or matches a token in the server's history table. ***Boolean***
<SYNCERROR>	V2 only	Optional error code, ***N-6***
<PMTMAILTRNRS>		Payment-mail transactions (0 or more)
</PMTMAILTRNRS>		
</PMTMAILSYNCRS>		

12.9 Payee Lists

Payments-system servers store lists of payees set up for payment by each user. Some systems require this before the user can issue a payment to a payee. In other payment systems, this feature enables the sharing of payee entry among multiple clients, and simplifies server payee maintenance.

A server-assigned payee list-entry ID identifies entries in the payee list. The following set of messages allows clients to obtain this list of payees. Users can add, modify, and delete individual entries in the list. The request for the list is synchronized, so that multiple clients can use the list.

Creating a payee:

Client Sends	*Server Responds*
Server-assigned payee identifier, or payee billing address User's account number with the payee	
	Payee address Standard payee information

Modifying a payee:

Client Sends	*Server Responds*
Server-assigned payee identifier User's account number with the payee Information to change: payee name address city state postal code phone number	

Client Sends	Server Responds
new payee account #	Extended payee information if payee is a standard payee or has non-default processing lead times Acknowledgment or Error

Deleting a payee:

Client Sends	Server Responds
Server-assigned payee identifier User's account number with the payee	Acknowledgment or Error

12.9.1 Adding a Payee to the Payee List

The user can use the Payee Request to add a payee to the server payee list. The server responds with a Payee Response, which can contain a complete billing address for the payee, or if the payee is a standard payee, the lead-time and payee name values.

12.9.1.1 Payee Request <PAYEERQ>

The <PAYEERQ> requests the addition of a payee entry to the server's payee list. Note that the user can use a <PMTRQ> to simultaneously set up a payee. Open Financial Exchange does not require the client to send a <PAYEERQ> before making an initial <PMTRQ> to a payee.

The <PAYEERQ> request must appear within a <PAYEETRNRQ> transaction wrapper.

Tag	Version	Description
<PAYEERQ>		Payee-request aggregate
<PAYEEID>	V1 only	Server-assigned payee identifier. Either <PAYEEID> or <PAYEE> but not both may be sent. ***SRVRTID***
<PAYEEID2>	V2 only	Server-assigned payee identifier. Either <PAYEEID2> or <PAYEE> but not both may be sent. ***SRVRTID2***
<PAYEE>	V1 only	Complete payee billing information, see section 12.5.2.1. Either <PAYEEID> or <PAYEE> but not both may be sent
</PAYEE>		
<PAYEE2>	V2 only	Complete payee billing information, see section 12.5.2.1. Either <PAYEEID2> or <PAYEE> but not both may be sent
</PAYEE2>		
<BANKACCTTO>		The destination bank account, specified in countries that pay using transfers. The <PAYEE> (above) must also be specified.
</BANKACCTTO>		
<PAYACCT>		User's account number(s) with the payee (0 or more), ***A-32***
</PAYEERQ>		

12.9.1.2 Payee Response <PAYEERS>

The server sends the Payee Response in response to a Payee Request. It contains the full billing information for the payee if it is not a standard payee. Otherwise, it contains the standard payee information, including lead time and payee name. If the server identifies the payee as having an assigned payee ID, then the server will include the <EXTDPAYEE> aggregate in the response.

If the response indicates that the payee does not have an assigned <PAYEEID>, the client should specify the full billing address <PAYEE> information in subsequent payment requests <PMTRQ> to the payee if the payee is being modified, otherwise the <PAYEELSTID> is used in lieu of the <PAYEE> aggregate.

If the response indicates that the payee does have a <PAYEEID>, then the client should use the <PAYEEID> for making payments to that payee.

The <PAYEERS> response must appear within a <PAYEETRNRS> transaction wrapper.

Tag	*Version*	*Description*
<PAYEERS>		Payee-response aggregate
<PAYEELSTID>	V1 only	Server-assigned record ID for this payee record, ***SRVRTID***
<PAYEELSTID2>	V2 only	Server-assigned record ID for this payee record, ***SRVRTID2***
<PAYEE>	V1 only	Complete payee billing information, see section 12.5.2.1
</PAYEE>		
<PAYEE2>	V2 only	Complete payee billing information, see section 12.5.2.1.
</PAYEE2>		
<BANKACCTTO>		The destination bank account, specified in countries that pay using transfers. The <PAYEE> (above) must also be specified.
</BANKACCTTO>		
<EXTDPAYEE>		Extended payee information, see section 12.5.2.3
</EXTDPAYEE>		
<PAYACCT>		User's account number(s) with the payee (0 or more), ***A-32***
</PAYEERS>		

12.9.1.3 Status Codes

Code	*Meaning*
0	Success (INFO)
2000	General error (ERROR)
2001	Invalid account (ERROR)
2002	General account error (ERROR)
2009	Destination account not found (ERROR)
2010	Destination account closed (ERROR)
2011	Destination account not authorized (ERROR)
2019	Duplicate request (ERROR)
10501	Invalid payee (ERROR)
10502	Invalid payee address (ERROR)
10503	Invalid payee account number (ERROR)

Code	Meaning
10511	Invalid payee city (ERROR)
10512	Invalid payee state (ERROR)
10513	Invalid payee postal code (ERROR)

12.9.2 Payee Modification

The Payee Modification Request allows the client to make changes to payee entries in the server's payee list. Changes in the server's payee list are not propagated to already created/spawned pending payments. Recurring payments created after the payee modification will use the updated information from the server's payee list as modified by the Payee Modification request.

12.9.2.1 Request <PAYEEMODRQ>

The client sends the Payee Modification Request to request changes to an existing payee list entry. The <PAYEE> aggregate must specify the changed and unchanged payee information. Absence of a <PAYACCT> in a <PAYEEMODRQ> could be interpreted as an implicit disassociation of the <PAYACCT> with the payee. Presence or absence of a <PAYACCT> does not imply selective <PAYEE> aggregate changes for the same <PAYEELSTID> as referenced by more than one <PAYACCT>. The <PAYEEMODRQ> request must appear within a <PAYEETRNRQ> transaction wrapper.

Tag	Version	Description
<PAYEEMODRQ>		Modification-request aggregate
<PAYEELSTID>	V1 only	Server-assigned record ID for this payee record, ***A-12***
<PAYEELSTID2>	V2 only	Server-assigned record ID for this payee record, ***SRVRTID2***
<PAYEE>	V1 only	Payee information to modify
</PAYEE>		
<PAYEE2>	V2 only	Payee information to modify
</PAYEE2>		
<BANKACCTTO>		Destination account for countries that pay using transfers (<PAYEE> required)
</BANKACCTTO>		
<PAYACCT>		Payer account number(s) with the payee (0 or more), ***A-32***
</PAYEEMODRQ>		

12.9.2.2 Response <PAYEEMODRS>

The server returns a Payee Modification Response in reply to a Payee Modification Request.

When a server-initiated change occurs to the extended payee information for a payee (for example a change in the payee's lead-time), the server can include this information in the <EXTDPAYEE> of the response.

If a server-initiated response indicates either that a payee now has a payee ID, or no longer has one, then the client should use the appropriate form of designating the payee in any future payment requests <PMTRQ> to that payee.

The <PAYEEMODRS> response must appear within a <PAYEETRNRS> transaction wrapper.

Tag	Version	Description
<PAYEEMODRS>		Modification-response aggregate

Tag	Version	Description
<PAYEELSTID>	V1 only	Server-assigned record ID for this payee record, ***A-12***
<PAYEELSTID2>	V2 only	Server-assigned record ID for this payee record, ***SRVRTID2***
<PAYEE>	V1 only	Payee information that was modified, see section 12.5.2.1
</PAYEE>		
<PAYEE2>	V2 only	Payee information that was modified, see section 12.5.2.1
</PAYEE2>		
<BANKACCTTO>		Destination account for countries that pay bills using transfers (<PAYEE> required as well)
</BANKACCTTO>		
<PAYACCT>		Payer's account number(s) with the payee (0 or more), ***A-32***
<EXTDPAYEE>		Extended payee information, see section 12.5.2.3
</EXTDPAYEE>		
</PAYEEMODRS>		

12.9.2.3 Status Codes

Code	Meaning
0	Success (INFO)
2000	General error (ERROR)
2001	Invalid account (ERROR)
2002	General account error (ERROR)
2009	Destination account not found (ERROR)
2010	Destination account closed (ERROR)
2011	Destination account not authorized (ERROR)
2019	Duplicate request (ERROR)
10501	Invalid payee (ERROR)
10502	Invalid payee address (ERROR)
10503	Invalid payee account number (ERROR)
10510	Invalid payee ID (ERROR)
10511	Invalid payee city (ERROR)
10512	Invalid payee state (ERROR)
10513	Invalid payee postal code (ERROR)
10515	Payee not modifiable by client (ERROR)

12.9.3 Payee Deletion

The Payee Deletion Request allows a client to delete a payee entry from the server's list of the user's payees. To delete specific <PAYACCT> associations with a payee, clients should use the <PAYEELSTID> combined with absent <PAYACCTS> via a <PAYEEMODRQ>.

The Payee delete request does not cancel payments that are pending at the time of the payee's deletion. References to pending payments subsequent to a payee's deletion pose issues regarding <PAYEELSTID> assignment at both the client and server levels. Therefore, it is suggested that the client disallow payee deletes if there are pending payments/models.

12.9.3.1 Request <PAYEEDELRQ>

The <PAYEEDELRQ> requests the deletion of a payee entry.

The <PAYEEDELRQ> request must appear within a <PAYEETRNRQ> transaction wrapper.

Tag	*Version*	*Description*
<PAYEEDELRQ>		Deletion-request aggregate
<PAYEELSTID>	V1 only	Server-assigned record ID for this payee record, ***SRVRTID***
<PAYEELSTID2>	V2 only	Server-assigned record ID for this payee record, ***SRVRTID2***
</PAYEEDELRQ>		

12.9.3.2 Response <PAYEEDELRS>

The server sends the Payee Deletion Response <PAYEEDELRS> in response to a Payee Deletion Request <PAYEEDELRQ>.

The <PAYEEDELRS> response must appear within a <PAYEETRNRS> transaction wrapper.

Tag	*Version*	*Description*
<PAYEEDELRS>		Deletion-response aggregate
<PAYEELSTID>	V1 only	Server-assigned record ID for this payee record, ***SRVRTID***
<PAYEELSTID2>	V2 only	Server-assigned record ID for this payee record, ***SRVRTID2***
</PAYEEDELRS>		

12.9.3.3 Status Codes

Code	*Meaning*
0	Success (INFO)
2000	General error (ERROR)
2019	Duplicate request (ERROR)
10519	Invalid payee list ID (ERROR)

12.9.4 Payee List Synchronization

This set of messages allows clients to obtain a list of payees stored on the server that it has configured for use in payments. In a "pay some" system, users are sometimes required to explicitly configure the payees to whom the system will make payments. This can be done by means of a telephone call to the payments provider or through some other interface. The client can then use this message set to obtain the user's list of configured payees. In other systems, the payments provider can elect to store a list of all payees that the user has paid. This is a convenience to the client. It allows payees set up on one client to be accessible from a user's other clients. The support of this message set is optional, but can be required by "pay some" providers that rely on this functionality.

12.9.4.1 Request <PAYEESYNCRQ>

Tag	*Version*	*Description*
<PAYEESYNCRQ>		Payee-list-request aggregate
Client synchronization option; <TOKEN>, <TOKEN2>, <TOKENONLY>, or <REFRESH>		
<TOKEN>	V1 only	Previous value of <TOKEN> received for this type of synchronization request from server; 0 for first-time requests; ***token***
<TOKEN2>	V2 only	Previous value of <TOKEN2> received for this type of synchronization request from server; 0 for first-time requests; ***token2***
<TOKENONLY>		Request for just the current <TOKEN> without the history, ***Boolean***
<REFRESH>		Request for refresh of current state, ***Boolean***
<REJECTIFMISSING>		If Y, do not process requests if client <TOKEN> is out of date, ***Boolean***
<PAYEETRNRQ>		Payee transactions (0 or more)
</PAYEETRNRQ>		
</PAYEESYNCRQ>		

12.9.4.2 Response <PAYEESYNCRS>

Tag	*Version*	*Description*
<PAYEESYNCRS>		Payee-list-request aggregate
<TOKEN>	V1 only	New synchronization token, ***token***
<TOKEN2>	V2 only	New synchronization token, ***token2***
<LOSTSYNC>		Y if the token in the synchronization request is older than the earliest entry in the server's history table. In this case, some responses have been lost. N if the token in the synchronization request is newer than or matches a token in the server's history table. ***Boolean***
<SYNCERROR>	V2 only	Optional error code, ***N-6***
<PAYEETRNRS>		Payee transactions (0 or more)
</PAYEETRNRS>		
</PAYEESYNCRS>		

12.9.4.3 Status Codes

Code	*Meaning*

Code	Meaning
0	Success (INFO)
2000	General error (ERROR)

12.10 Data Synchronization for Payments

Users of Open Financial Exchange Payments need to be able to obtain the current status of transactions previously sent to the server for processing. For example, once the user schedules a payment and the payment date has passed, the user might want to verify that the server made the payment as directed. Additionally, Open Financial Exchange allows for interactions with the server through multiple clients. This means, for example, that the user can perform some transactions from a home PC and others from an office computer with each session incorporating the activities performed on the other.

In order to accomplish these tasks, the client uses a synchronization scheme to insure that it has an accurate copy of the server data that is relevant to the client application. The intent of this scheme is to address three scenarios in which the client might lose synchronization with the server:

- A transaction has changed its state based on processing actions on the server. For example, a scheduled payment has passed its due date and has been paid or rejected.
- Transactions relevant to the client's application state have been added, deleted, or modified by a second client. For example, a user might enter or change transactions from more than one PC or application.
- A communications session between the client and server was interrupted or completed abnormally. As a result the client does not have responses from the server indicating that all the transactions were received and processed.

12.10.1 Payment Synchronization

12.10.1.1 Request <PMTSYNCRQ>

Tag	Version	Description
<PMTSYNCRQ>		Synchronization-request aggregate
Client synchronization option; <TOKEN>, <TOKEN2>, <TOKENONLY>, or <REFRESH>		
<TOKEN>	V1 only	Previous value of <TOKEN> received for this type of synchronization request from server; 0 for first-time requests; ***token***
<TOKEN2>	V2 only	Previous value of <TOKEN2> received for this type of synchronization request from server; 0 for first-time requests; ***token2***
<TOKENONLY>		Request for just the current <TOKEN> without the history, ***Boolean***
<REFRESH>		Request for refresh of current state, ***Boolean***
<REJECTIFMISSING>		If Y, do not process requests if client <TOKEN> is out of date, ***Boolean***
<BANKACCTFROM>		Opening tag for account from aggregate, see section 11.3.1
</BANKACCTFROM>		
<PMTTRNRQ>		Payment transactions (0 or more)
</PMTTRNRQ>		
</PMTSYNCRQ>		

12.10.1.2 Response <PMTSYNCRS>

Tag	*Version*	*Description*
<PMTSYNCRS>		Synchronization-response aggregate
<TOKEN>	V1 only	New synchronization token, ***token***
<TOKEN2>	V2 only	New synchronization token, ***token2***
<LOSTSYNC>		Y if the token in the synchronization request is older than the earliest entry in the server's history table. In this case, some responses have been lost. N if the token in the synchronization request is newer than or matches a token in the server's history table. ***Boolean***
<SYNCERROR>	V2 only	Optional error code, ***N-6***
<BANKACCTFROM>		Opening tag for account from aggregate, see section 11.3.1
</BANKACCTFROM>		
<PMTTRNRS>		Payment transactions (0 or more)
</PMTTRNRS>		
</PMTSYNCRS>		

12.10.2 Recurring Payment Synchronization

12.10.2.1 Request <RECPMTSYNCRQ>

Tag	*Version*	*Description*
<RECPMTSYNCRQ>		Synchronization-request aggregate
Client synchronization option; <TOKEN>, <TOKEN2>, <TOKENONLY>, or <REFRESH>		
<TOKEN>	V1 only	Previous value of <TOKEN> received for this type of synchronization request from server; 0 for first-time requests; ***token***
<TOKEN2>	V2 only	Previous value of <TOKEN2> received for this type of synchronization request from server; 0 for first-time requests; ***token2***
<TOKENONLY>		Request for just the current <TOKEN> without the history, ***Boolean***
<REFRESH>		Request for refresh of current state, ***Boolean***
<REJECTIFMISSING>		If Y, do not process requests if client <TOKEN> is out of date, ***Boolean***
<BANKACCTFROM>		Opening tag for account from aggregate, see section 11.3.1
</BANKACCTFROM>		
<RECPMTTRNRQ>		Recurring-payment transactions (0 or more)
</RECPMTTRNRQ>		
</RECPMTSYNCRQ>		

12.10.2.2 Response <RECPMTSYNCRS>

Tag	*Version*	*Description*
<RECPMTSYNCRS>		Synchronization-response aggregate
<TOKEN>	V1 only	New synchronization token, ***token***

Tag	Version	Description
<TOKEN2>	V2 only	New synchronization token, ***token2***
<LOSTSYNC>		Y if the token in the synchronization request is older than the earliest entry in the server's history table. In this case, some responses have been lost. N if the token in the synchronization request is newer than or matches a token in the server's history table. ***Boolean***
<SYNCERROR>	V2 only	Optional error code, ***N-6***
<BANKACCTFROM>		Opening tag for account from aggregate, see section 11.3.1
</BANKACCTFROM>		
<RECPMTTRNRS>		Recurring-payment transactions (0 or more)
</RECPMTTRNRS>		
</RECPMTSYNCRS>		

12.10.2.3 Status Codes

Code	Meaning
0	Success (INFO)
2000	General error (ERROR)
2002	General account error (ERROR)
2003	Account not found (ERROR)
2004	Account closed (ERROR)
2005	Account not authorized (ERROR)

12.10.3 Discussion

This section describes specific synchronization processing for the Open Financial Exchange Payments functions. Chapter 6, “Data Synchronization,” provides a more extensive discussion of the Open Financial Exchange synchronization mechanism.

The client follows the steps below to synchronize:

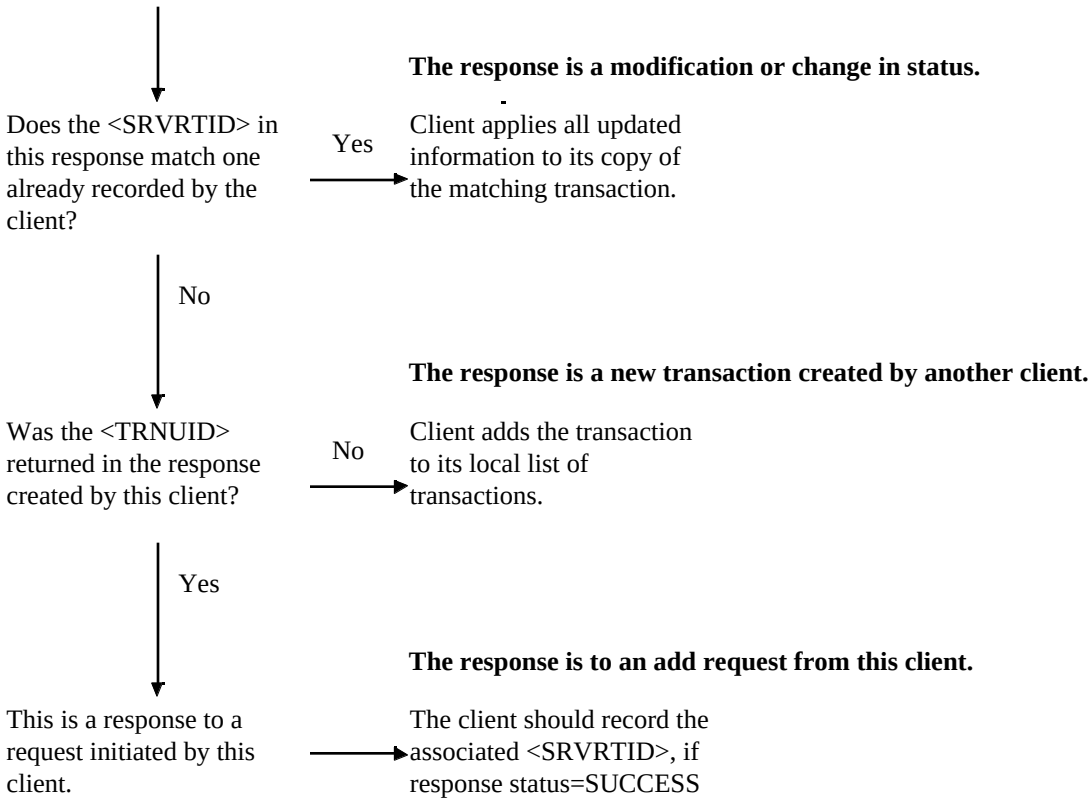
1. The client sends a <PMTSYNCRQ> and/or <RECPMTSYNCRQ> containing the token it has stored from its last successful synchronization (or the special initial token value).
2. The client processes the <PMTSYNCRS> and/or <RECPMTSYNCRS> response from the server.

When the client has requested the server to add a transaction, a response that contains a <TRNUID> matching a transaction originally sent by the client—for which the client has not recorded an associated <SRVRTID>—is the normal scenario. This scenario could also occur if the server response did not reach the client in the previous session. In either case, the client should add these server IDs to their associated transactions at this point.

If the client previously recorded the <SRVRTID>, this response is a change in status or in the contents of the transaction. The request might have originated from this client, another client, or might be the result of server processing.

If the <TRNUID> does not match any transaction known to the client, a second client initiated this transaction. In rarer cases the response might be a transaction initially requested by this client, for which the client has lost its record; for example, the client has reverted to a backup.

The diagram below describes the processing and interpretation of <SRVRTID> and <TRNUID> identifiers by the client:



After the server has processed all the synchronization responses, the client scans its database of transactions to verify that they have all been assigned a <SRVRTID>. Any transactions missing this identifier were never received by the server and should be resent (using the originally assigned <TRNUID> to avoid duplicate requests). Additionally, the client should record the <TOKEN> received in the response.

12.11 Message Sets and Profile

Open Financial Exchange separates messages that the client and server send into groups called message sets. Each financial institution defines the message sets that a particular institution will support. Currently, all the payment messages described in this chapter fall into a single message set.

The message set contains options and attributes that allow a financial institution to customize its use of Open Financial Exchange. The options and attributes are defined in the profile as part of the message set definition. Each set of options and attributes appears within an aggregate that is specific to a message set. Specifically, all of the options and attributes that pertain to payments are contained within <BILLPAYMSGSETV1> and <BILLPAYMSGSETV2>.

12.11.1 Bill Pay Message Sets and Messages

12.11.1.1 Bill Pay Message Set Request Messages

Message Set	*Messages*
<BILLPAYMSGSET>	
<BILLPAYMSGSETV1>	
<BILLPAYMSGSRQV1>	PMTTRNRQ
	PMTRQ
	PMTMODRQ
	PMTCANCRQ
	RECPMTTRNRQ
	RECPMTRQ
	RECPMTMODRQ
	RECPMTCANCRQ
	PAYEETRNRQ
	PAYEERQ
	PAYEEMODRQ
	PAYEEDELRQ
	PMTINQTRNRQ
	PMTINQRQ
	PMTMAILTRNRQ
	PMTMAILRQ
	PMTSYNCRQ
	RECPMTSYNCRQ
	PAYEESYNCRQ
	PMTMAILSYNCRQ
</BILLPAYMSGSRQV1>	
</BILLPAYMSGSETV1>	
<BILLPAYMSGSETV2>	
<BILLPAYMSGSRQV2>	PMTTRNRQ

Message Set	Messages
	PMTRQ
	PMTMODRQ
	PMTCANCRQ
	RECPMTTRNRQ
	RECPMTRQ
	RECPMTMODRQ
	RECPMTCANCRQ
	PAYEETRNRQ
	PAYEERQ
	PAYEEMODRQ
	PAYEEDELRQ
	PMTINQTRNRQ
	PMTINQRQ
	PMTMAILTRNRQ
	PMTMAILRQ
	PMTSYNCRQ
	RECPMTSYNCRQ
	PAYEESYNCRQ
	PMTMAILSYNCRQ
</BILLPAYMSGSRQV2>	
</BILLPAYMSGSETV2>	
</BILLPAYMSGSET>	

12.11.1.2 Bill Pay Message Set Response Messages

Message Set	Messages
<BILLPAYMSGSET>	
<BILLPAYMSGSETV1>	
<BILLPAYMSGSRSV1>	PMTTRNRS
	PMTRS
	PMTMODRS
	PMTCANCRS
	RECPMTTRNRS
	RECPMTRS
	RECPMTMODRS
	RECPMTCANCRS
	PAYEETRNRS
	PAYEERS
	PAYEEMODRS
	PAYEEDELRS
	PMTINQTRNRS
	PMTINQRS

Message Set	*Messages*
	PMTMAILTRNRS
	PMTMAILRS
	PMTSYNCRS
	RECPMTSYNCRS
	PAYEESYNCRS
	PMTMAILSYNCRS
</BILLPAYMSGSRSV1>	
</BILLPAYMSGSETV1>	
<BILLPAYMSGSETV2>	
<BILLPAYMSGSRSV2>	PMTTRNRS
	PMTRS
	PMTMODRS
	PMTCANCRS
	RECPMTTRNRS
	RECPMTRS
	RECPMTMODRS
	RECPMTCANCRS
	PAYEETRNRS
	PAYEERS
	PAYEEMODRS
	PAYEEDELRS
	PMTINQTRNRS
	PMTINQRS
	PMTMAILTRNRS
	PMTMAILRS
	PMTSYNCRS
	RECPMTSYNCRS
	PAYEESYNCRS
	PMTMAILSYNCRS
</BILLPAYMSGSRSV2>	
</BILLPAYMSGSETV2>	
</BILLPAYMSGSET>	

12.11.2 Bill Pay Message Set Profile <BILLPAYMSGSET>

Tag	*Description*
<BILLPAYMSGSET>	
<BILLPAYMSGSETV1>	Version 1 of bill pay message set
<MSGSETCORE>	

Tag	Description
</MSGSETCORE>	
<DAYSWITH>	Offset to withdrawal date, such that (DTDUE – DAYSTOPAY) + (DAYSWITH) determines the date on which the funds are withdrawn from the user's account. ***N-3***
	Note: If the value of <DAYSWITH> is "-1," then the withdrawal date is the same as the payment date (<DTDUE>).
<DFLTDAYSTOPAY>	Default number of days to pay by check (except by transfer), ***N-3***
	Can be overridden for each payee, by <DAYSTOPAY> in the <EXTDPAYEE> aggregate, see section 12.5.2.3
<XFERDAYSWITH>	Number of days before processing date that funds are withdrawn for payment by transfer, ***N-3***
<XFERDFLTDAYSTOPAY>	Default number of days to pay by transfer, ***N-3***
	Can be overridden for each payee, by <DAYSTOPAY> in the <EXTDPAYEE> aggregate, see section 12.5.2.3
<PROCDAYSOFF>	Days of week that no processing occurs; 0 or more of (MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, SATURDAY, SUNDAY). <PROCDAYSOFF> indicate days to exclude when calculating dates that utilize other bill payment bits, such as <DAYSWITH> and <DFLTDAYSTOPAY> values.
<PROCENDTM>	Time of day that day's processing ends, ***time***
<MODELWND>	Model window; the number of days before a recurring transaction is scheduled to be processed that it is instantiated on the system, ***N-3***
<POSTPROCWND>	Number of days after a transaction is processed that it is accessible for status inquiries, ***N-3***
<STSVIAMODS>	If Y, server supports communication of server-initiated payment status changes by means of the PMTMODRS message
<PMTBYADDR>	The payment provider supports payments to payees identified by billing address, that is, the PAYEE aggregate, ***Boolean***
<PMTBYXFER>	The payment provider supports payments to payees identified by destination account, ***Boolean***
<PMTBYPAYEEID>	The payment provider supports payments to payees identified by a user-supplied payee ID, ***Boolean***
<CANADDPAYEE>	User can add payees. if no, the user is restricted to payees added to the user's payee list by the payment system, ***Boolean***
<HASEXTDPMT>	Supports the EXTDPMT business payment aggregate, ***Boolean***
<CANMODPMTS>	Permits modifications to payments, that is PMTMODRQ, ***Boolean***
<CANMODMDLS>	Permits modifications to models, that is REQPMTMODRQ, ***Boolean***
<DIFFFIRSTPMT>	Support for specifying a different amount for the first payment generated by a model, ***Boolean***
<DIFFLASTPMT>	Support for specifying a different amount for the last payment generated by a model, ***Boolean***
</BILLPAYMSGSETV1>	
<BILLPAYMSGSETV2>	Version 2 of bill pay message set
<MSGSETCORE>	
</MSGSETCORE>	
<DAYSWITH>	Offset to withdrawal date, such that (DTDUE – DAYSTOPAY) + (DAYSWITH) determines the date on which the funds are withdrawn from the user's account. ***N-3***
	Note: If the value of <DAYSWITH> is "-1," then the withdrawal date is the same as the payment date (<DTDUE>).

Tag	Description
<DFLTDAYSTOPAY>	Default number of days to pay by check (except by transfer), ***N-3***
	Can be overridden for each payee, by <DAYSTOPAY> in the <EXTDPAYEE> aggregate, see section 12.5.2.3
<XFERDAYSWITH>	Number of days before processing date that funds are withdrawn for payment by transfer, ***N-3***
<XFERDFLTDAYSTOPAY>	Default number of days to pay by transfer, ***N-3***
	Can be overridden for each payee, by <DAYSTOPAY> in the <EXTDPAYEE> aggregate, see section 12.5.2.3
	Note: If the value of <XFERDFLTDAYSTOPAY> is –1, then <DTDUE> is interpreted as an execution date (the date that the payer's bank will begin processing the payment) rather than a due date.
<PROCDAYSOFF>	Days of week that no processing occurs; 0 or more of (MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, SATURDAY, SUNDAY)
<PROCENDTM>	Time of day that day's processing ends, ***time***
<MODELWND>	Model window; the number of days before a recurring transaction is scheduled to be processed that it is instantiated on the system, ***N-3***
<POSTPROCWND>	Number of days after a transaction is processed that it is accessible for status inquiries, ***N-3***
<STSVIAMODS>	If Y, server supports communication of server-initiated payment status changes by means of the PMTMODRS message
<PMTBYADDR>	The payment provider supports payments to payees identified by billing address, that is, the PAYEE aggregate, ***Boolean***
<PMTBYXFER>	The payment provider supports payments to payees identified by destination account, ***Boolean***
<PMTBYPAYEEID>	The payment provider supports payments to payees identified by a user-supplied payee ID, ***Boolean***
<CANADDPAYEE>	User can add payees. if no, the user is restricted to payees added to the user's payee list by the payment system, ***Boolean***
<HASEXTDPMT>	Supports the EXTDPMT business payment aggregate, ***Boolean***
<CANMODPMTS>	Permits modifications to payments, that is PMTMODRQ, ***Boolean***
<CANMODMDLS>	Permits modifications to models, that is REQPMTMODRQ, ***Boolean***
<DIFFFIRSTPMT>	Support for specifying a different amount for the first payment generated by a model, ***Boolean***
<DIFFLASTPMT>	Support for specifying a different amount for the last payment generated by a model, ***Boolean***
<NEEDTANPMT>	A <TAN> is required for user authentication in the transaction wrapper of a request to create, modify or cancel a payment or model (see section 2.4.6), ***Boolean***
	For <COUNTRY> = USA, this will always be set to N.
<NEEDTANPAYEE>	A <TAN> is required for user authentication in the transaction wrapper of a request to create, modify or delete a payee (see section 2.4.6), ***Boolean***
	For <COUNTRY> = USA, this will always be set to N.
<SUPPORTDTAVAIL>	Support for specifying a payment value date using the <DTAVAIL> tag in the <PMTINFO> aggregate (see section 12.5.2), ***Boolean***
	For <COUNTRY> = USA, this will always be set to N.
</BILLPAYMSGSETV2>	
</BILLPAYMSGSET>	

12.12 Examples

12.12.1 Scheduling a Payment

Create a payment to "J.C. Counts" for $123.45 to be paid on September 1,1997 using funds in a checking account:

```
<!-- payment example 1 -->

<OFX>
   <SIGNONMSGSRQV1>
      <SONRQ>
         <DTCLIENT>19961029101000
         <USERID>123-45-6789
         <USERPASS>MyPassword
         <LANGUAGE>ENG
         <FI>
            <ORG>NCH
            <FID>12321
         </FI>
         <APPID>MyApp
         <APPVER>0700
      </SONRQ>
   </SIGNONMSGSRQV1>
   <BILLPAYMSGSRQV1>
      <PMTTRNRQ>
         <TRNUID>1001
         <PMTRQ>
            <PMTINFO>
               <BANKACCTFROM>
                  <BANKID>123432123
                  <ACCTID>516273
                  <ACCTTYPE>CHECKING
               </BANKACCTFROM>
               <TRNAMT>123.45
               <PAYEE>
                  <NAME>J. C. Counts
                  <ADDR1>100 Main St.
                  <CITY>Turlock
                  <STATE>CA
                  <POSTALCODE>90101
                  <PHONE>415.987.6543
               </PAYEE>
               <PAYACCT>10101
               <DTDUE>19971001
               <MEMO>payment #3
            </PMTINFO>
         </PMTRQ>
      </PMTTRNRQ>
   </BILLPAYMSGSRQV1>
</OFX>
```

The server responds indicating that it will make the payment on the date requested and that the payee is a standard payee:

```
<OFX>
   <SIGNONMSGSRSV1>
      <SONRS>
         <STATUS>
            <CODE>0
```

```
            <SEVERITY>INFO
         </STATUS>
         <DTSERVER>19961029101003
         <LANGUAGE>ENG
         <DTPROFUP>19961029101003
         <DTACCTUP>19961029101003
      </SONRS>
   </SIGNONMSGSRSV1>
   <BILLPAYMSGSRSV1>
      <PMTTRNRS>
         <TRNUID>1001
         <STATUS>
            <CODE>0
            <SEVERITY>INFO
         </STATUS>
         <PMTRS>
            <SRVRTID>1030155
            <PAYEELSTID>123214
            <CURDEF>USD
            <PMTINFO>
               <BANKACCTFROM>
                  <BANKID>123432123
                  <ACCTID>516273
                  <ACCTTYPE>CHECKING
               </BANKACCTFROM>
               <TRNAMT>123.45
               <PAYEE>
                  <NAME>J. C. Counts
                  <ADDR1>100 Main St.
                  <CITY>Turlock
                  <STATE>CA
                  <POSTALCODE>90101
                  <PHONE>415.987.6543
               </PAYEE>
               <PAYACCT>10101
               <DTDUE>19971001
               <MEMO>payment #3
            </PMTINFO>
            <EXTDPAYEE>
               <PAYEEID>9076
               <IDSCOPE>USER
               <NAME>J. C. Counts
               <DAYSTOPAY>3
            </EXTDPAYEE>
            <CHECKNUM>20111
           <PMTPRCSTS>
             <PMTPRCCODE>WILLPROCESSON
             <DTPMTPRC>19971001
           </PMTPRCSTS>
         </PMTRS>
      </PMTTRNRS>
   </BILLPAYMSGSRSV1>
</OFX>
```

Create a second payment to the payee, using the payee ID returned in the previous example:

```
<!-- payment example 2 -->

<OFX>
   <SIGNONMSGSRQV1>
      <SONRQ>
         <DTCLIENT>19961029101000
         <USERID>123-45-6789
         <USERPASS>MyPassword
         <LANGUAGE>ENG
```

```
         <FI>
            <ORG>NCH
            <FID>12321
         </FI>
         <APPID>MyApp
         <APPVER>0700
      </SONRQ>
   </SIGNONMSGSRQV1>
   <BILLPAYMSGSRQV1>
      <PMTTRNRQ>
         <TRNUID>1001
         <PMTRQ>
            <PMTINFO>
               <BANKACCTFROM>
                  <BANKID>123432123
                  <ACCTID>516273
                  <ACCTTYPE>CHECKING
               </BANKACCTFROM>
               <TRNAMT>123.45
               <PAYEEID>9076
               <PAYACCT>10101
               <DTDUE>19971101
               <MEMO>Payment #4
            </PMTINFO>
         </PMTRQ>
      </PMTTRNRQ>
   </BILLPAYMSGSRQV1>
</OFX>
```

The server responds indicating that it will make the payment on the date requested:

```
<OFX>
   <SIGNONMSGSRSV1>
      <SONRS>
         <STATUS>
            <CODE>0
            <SEVERITY>INFO
         </STATUS>
         <DTSERVER>19961029101003
         <LANGUAGE>ENG
         <DTPROFUP>19961029101003
         <DTACCTUP>19961029101003
      </SONRS>
   </SIGNONMSGSRSV1>
   <BILLPAYMSGSRSV1>
      <PMTTRNRS>
         <TRNUID>1001
         <STATUS>
            <CODE>0
            <SEVERITY>INFO
         </STATUS>
         <PMTRS>
            <SRVRTID>1068405
            <PAYEELSTID>123432
            <CURDEF>USD
            <PMTINFO>
               <BANKACCTFROM>
                  <BANKID>123432123
                  <ACCTID>516273
                  <ACCTTYPE>CHECKING
               </BANKACCTFROM>
               <TRNAMT>123.45
               <PAYEEID>9076
               <PAYACCT>10101
               <DTDUE>19971101
```

```
            <MEMO>payment #4
         </PMTINFO>
         <EXTDPAYEE>
            <PAYEEID>9076
            <IDSCOPE>USER
            <NAME>J. C. Counts
            <DAYSTOPAY>3
         </EXTDPAYEE>
        <PMTPRCSTS>
          <PMTPRCCODE>WILLPROCESSON
          <DTPMTPRC>19971101
        </PMTPRCSTS>
      </PMTRS>
   </PMTTRNRS>
</BILLPAYMSGSRSV1>
</OFX>
```

12.12.2 Modifying a Payment

Change the amount of a payment to $125.99

```
<OFX>
   <SIGNONMSGSRQV1>
      <SONRQ>
         <DTCLIENT>19961029101000
         <USERID>123-45-6789
         <USERPASS>MyPassword
         <LANGUAGE>ENG
         <FI>
            <ORG>NCH
            <FID>12321
         </FI>
         <APPID>MyApp
         <APPVER>0700
      </SONRQ>
   </SIGNONMSGSRQV1>
   <BILLPAYMSGSRQV1>
      <PMTTRNRQ>
         <TRNUID>1021
         <PMTMODRQ>
            <SRVRTID>1030776
            <PMTINFO>
               <BANKACCTFROM>
                  <BANKID>123432123
                  <ACCTID>516273
                  <ACCTTYPE>CHECKING
               </BANKACCTFROM>
               <TRNAMT>125.99                      <!-- changed amount -->
               <PAYEE>
                  <NAME>J. C. Counts
                  <ADDR1>100 Main St.
                  <CITY>Turlock
                  <STATE>CA
                  <POSTALCODE>90101
                  <PHONE>415.987.6543
               </PAYEE>
               <PAYACCT>10101
               <DTDUE>19971001
               <MEMO>payment #3
            </PMTINFO>
         </PMTMODRQ>
      </PMTTRNRQ>
   </BILLPAYMSGSRQV1>
```

```
</OFX>
```

Server response echoes the request:

```
<OFX>
   <SIGNONMSGSRSV1>
      <SONRS>
         <STATUS>
            <CODE>0
            <SEVERITY>INFO
         </STATUS>
         <DTSERVER>19961029101003
         <LANGUAGE>ENG
         <DTPROFUP>19961029101003
         <DTACCTUP>19961029101003
      </SONRS>
   </SIGNONMSGSRSV1>
   <BILLPAYMSGSRSV1>
      <PMTTRNRS>
         <TRNUID>1021
         <STATUS>
            <CODE>0
            <SEVERITY>INFO
         </STATUS>
         <PMTMODRS>
            <SRVRTID>1030776
            <PMTINFO>
               <BANKACCTFROM>
                  <BANKID>123432123
                  <ACCTID>516273
                  <ACCTTYPE>CHECKING
               </BANKACCTFROM>
               <TRNAMT>125.99                      <!-- changed amount -->
               <PAYEE>
                  <NAME>J. C. Counts
                  <ADDR1>100 Main St.
                  <CITY>Turlock
                  <STATE>CA
                  <POSTALCODE>90101
                  <PHONE>415.987.6543
               </PAYEE>
               <PAYACCT>10101
               <DTDUE>19971001
               <MEMO>payment #3
            </PMTINFO>
         </PMTMODRS>
      </PMTTRNRS>
   </BILLPAYMSGSRSV1>
</OFX>
```

Change the date of a payment to December 12, 1997

```
<OFX>
   <SIGNONMSGSRQV1>
      <SONRQ>
         <DTCLIENT>19961029101000
         <USERID>123-45-6789
         <USERPASS>MyPassword
         <LANGUAGE>ENG
         <FI>
            <ORG>NCH
            <FID>12321
         </FI>
         <APPID>MyApp
         <APPVER>0700
      </SONRQ>
```

```
   </SIGNONMSGSRQV1>
   <BILLPAYMSGSRQV1>
      <PMTTRNRQ>
         <TRNUID>32456
         <PMTMODRQ>
            <SRVRTID>1030776
            <PMTINFO>
               <BANKACCTFROM>
                  <BANKID>123432123
                  <ACCTID>516273
                  <ACCTTYPE>CHECKING
               </BANKACCTFROM>
               <TRNAMT>125.99
               <PAYEE>
                  <NAME>J. C. Counts
                  <ADDR1>100 Main St.
                  <CITY>Turlock
                  <STATE>CA
                  <POSTALCODE>90101
                  <PHONE>415.987.6543
               </PAYEE>
               <PAYACCT>10101
               <DTDUE>19971212                    <!-- changed date -->
               <MEMO>payment #3
            </PMTINFO>
         </PMTMODRQ>
      </PMTTRNRQ>
   </BILLPAYMSGSRQV1>
</OFX>
```

Server response echoes the request:

```
<OFX>
   <SIGNONMSGSRSV1>
      <SONRS>
         <STATUS>
            <CODE>0
            <SEVERITY>INFO
         </STATUS>
         <DTSERVER>19961029101003
         <LANGUAGE>ENG
         <DTPROFUP>19961029101003
         <DTACCTUP>19961029101003
      </SONRS>
   </SIGNONMSGSRSV1>
   <BILLPAYMSGSRSV1>
      <PMTTRNRS>
         <TRNUID>32456
         <STATUS>
            <CODE>0
            <SEVERITY>INFO
         </STATUS>
         <PMTMODRS>
            <SRVRTID>1030776
            <PMTINFO>
               <BANKACCTFROM>
                  <BANKID>123432123
                  <ACCTID>516273
                  <ACCTTYPE>CHECKING
               </BANKACCTFROM>
               <TRNAMT>125.99
               <PAYEE>
                  <NAME>J. C. Counts
                  <ADDR1>100 Main St.
                  <CITY>Turlock
```

```
                <STATE>CA
                <POSTALCODE>90101
                <PHONE>415.987.6543
             </PAYEE>
             <PAYACCT>10101
             <DTDUE>19971212                    <!-- changed date -->
             <MEMO>payment #3
          </PMTINFO>
       </PMTMODRS>
    </PMTTRNRS>
 </BILLPAYMSGSRSV1>
</OFX>
```

12.12.3 Canceling a Payment

Cancel a payment:

```
<OFX>
   <SIGNONMSGSRQV1>
      <SONRQ>
         <DTCLIENT>19971029101000
         <USERID>123-45-6789
         <USERPASS>MyPassword
         <LANGUAGE>ENG
         <FI>
            <ORG>NCH
            <FID>12321
         </FI>
         <APPID>MyApp
         <APPVER>0700
      </SONRQ>
   </SIGNONMSGSRQV1>
   <BILLPAYMSGSRQV1>
      <PMTTRNRQ>
         <TRNUID>54601
         <PMTCANCRQ>
            <SRVRTID>1030776
         </PMTCANCRQ>
      </PMTTRNRQ>
   </BILLPAYMSGSRQV1>
</OFX>
```

Server responds:

```
<OFX>
   <SIGNONMSGSRSV1>
      <SONRS>
         <STATUS>
            <CODE>0
            <SEVERITY>INFO
         </STATUS>
         <DTSERVER>19971029101003
         <LANGUAGE>ENG
         <DTPROFUP>19961029101003
         <DTACCTUP>19961029101003
      </SONRS>
   </SIGNONMSGSRSV1>
   <BILLPAYMSGSRSV1>
      <PMTTRNRS>
         <TRNUID>54601
         <STATUS>
            <CODE>0
```

```
            <SEVERITY>INFO
         </STATUS>
         <PMTCANCRS>
            <SRVRTID>1030776
         </PMTCANCRS>
      </PMTTRNRS>
   </BILLPAYMSGSRSV1>
</OFX>
```

12.12.4 Updating Payment Status

Update payment status:

```
<OFX>
   <SIGNONMSGSRQV1>
      <SONRQ>
         <DTCLIENT>19970129101000
         <USERID>123-45-6789
         <USERPASS>MyPassword
         <LANGUAGE>ENG
         <FI>
            <ORG>NCH
            <FID>12321
         </FI>
         <APPID>MyApp
         <APPVER>0700
      </SONRQ>
   </SIGNONMSGSRQV1>
   <BILLPAYMSGSRQV1>
      <PMTINQTRNRQ>
         <TRNUID>7865
         <PMTINQRQ>
            <SRVRTID>565321
         </PMTINQRQ>
      </PMTINQTRNRQ>
   </BILLPAYMSGSRQV1>
</OFX>
```

Server responds with updated status and check number:

```
<OFX>
   <SIGNONMSGSRSV1>
      <SONRS>
         <STATUS>
            <CODE>0
            <SEVERITY>INFO
         </STATUS>
         <DTSERVER>19961029101003
         <LANGUAGE>ENG
         <DTPROFUP>19961029101003
         <DTACCTUP>19961029101003
      </SONRS>
   </SIGNONMSGSRSV1>
   <BILLPAYMSGSRSV1>
      <PMTINQTRNRS>
         <TRNUID>7865
         <STATUS>
            <CODE>0
            <SEVERITY>INFO
         </STATUS>
         <PMTINQRS>
            <SRVRTID>565321
            <PMTPRCSTS>
               <PMTPRCCODE>PROCESSEDON
               <DTPMTPRC>19970201
```

```
            </PMTPRCSTS>
            <CHECKNUM>6017
         </PMTINQRS>
      </PMTINQTRNRS>
   </BILLPAYMSGSRSV1>
</OFX>
```

12.12.5 Scheduling a Recurring Payment

Create a recurring payment of 36 monthly payments of $395 to a standard payee. The first payment will be on November 15, 1997:

```
<OFX>
   <SIGNONMSGSRQV1>
      <SONRQ>
         <DTCLIENT>19961029101000
         <USERID>123-45-6789
         <USERPASS>MyPassword
         <LANGUAGE>ENG
         <FI>
            <ORG>NCH
            <FID>12321
         </FI>
         <APPID>MyApp
         <APPVER>0700
      </SONRQ>
   </SIGNONMSGSRQV1>
   <BILLPAYMSGSRQV1>
      <RECPMTTRNRQ>
         <TRNUID>1001
         <RECPMTRQ>
            <RECURRINST>
               <NINSTS>36
               <FREQ>MONTHLY
            </RECURRINST>
            <PMTINFO>
               <BANKACCTFROM>
                  <BANKID>555432180
                  <ACCTID>763984
                  <ACCTTYPE>CHECKING
               </BANKACCTFROM>
               <TRNAMT>395.00
               <PAYEEID>77810
               <PAYACCT>444-78-97572
               <DTDUE>19971115
               <MEMO>Auto loan payment
            </PMTINFO>
         </RECPMTRQ>
      </RECPMTTRNRQ>
   </BILLPAYMSGSRQV1>
</OFX>
```

Server response includes the assigned server transaction ID:

```
<OFX>
   <SIGNONMSGSRSV1>
      <SONRS>
         <STATUS>
            <CODE>0
            <SEVERITY>INFO
         </STATUS>
         <DTSERVER>19971029101003
         <LANGUAGE>ENG
         <DTPROFUP>19961029101003
         <DTACCTUP>19961029101003
```

```
      </SONRS>
   </SIGNONMSGSRSV1>
   <BILLPAYMSGSRSV1>
      <RECPMTTRNRS>
         <TRNUID>1001
         <STATUS>
            <CODE>0
            <SEVERITY>INFO
         </STATUS>
         <RECPMTRS>
            <RECSRVRTID>387687138
            <PAYEELSTID>27983
            <CURDEF>USD
            <RECURRINST>
               <NINSTS>36
               <FREQ>MONTHLY
            </RECURRINST>
            <PMTINFO>
               <BANKACCTFROM>
                  <BANKID>555432180
                  <ACCTID>763984
                  <ACCTTYPE>CHECKING
               </BANKACCTFROM>
               <TRNAMT>395.00
               <PAYEEID>77810
               <PAYACCT>444-78-97572
               <DTDUE>19971115
               <MEMO>Auto loan payment
            </PMTINFO>
         </RECPMTRS>
      </RECPMTTRNRS>
   </BILLPAYMSGSRSV1>
</OFX>
```

12.12.6 Modifying a Recurring Payment

Change the amount of a recurring payment:

```
<OFX>
   <SIGNONMSGSRQV1>
      <SONRQ>
         <DTCLIENT>19961029101000
         <USERID>123-45-6789
         <USERPASS>MyPassword
         <LANGUAGE>ENG
         <FI>
            <ORG>NCH
            <FID>12321
         </FI>
         <APPID>MyApp
         <APPVER>0700
      </SONRQ>
   </SIGNONMSGSRQV1>
   <BILLPAYMSGSRQV1>
      <RECPMTTRNRQ>
         <TRNUID>2001
         <RECPMTMODRQ>
            <RECSRVRTID>387687138
            <RECURRINST>
               <NINSTS>36
               <FREQ>MONTHLY
            </RECURRINST>
            <PMTINFO>
               <BANKACCTFROM>
```

```
                <BANKID>555432180
                <ACCTID>763984
                <ACCTTYPE>CHECKING
             </BANKACCTFROM>
             <TRNAMT>399.95                      <!-- changing amount -->
             <PAYEEID>77810
             <PAYACCT>444-78-97572
             <DTDUE>19971115
             <MEMO>Auto loan payment
          </PMTINFO>
          <MODPENDING>N
       </RECPMTMODRQ>
    </RECPMTTRNRQ>
 </BILLPAYMSGSRQV1>
</OFX>
```

Server responds:

```
<OFX>
   <SIGNONMSGSRSV1>
      <SONRS>
         <STATUS>
            <CODE>0
            <SEVERITY>INFO
         </STATUS>
         <DTSERVER>19971029101003
         <LANGUAGE>ENG
         <DTPROFUP>19961029101003
         <DTACCTUP>19961029101003
      </SONRS>
   </SIGNONMSGSRSV1>
   <BILLPAYMSGSRSV1>
      <RECPMTTRNRS>
         <TRNUID>2001
         <STATUS>
            <CODE>0
            <SEVERITY>INFO
         </STATUS>
         <RECPMTMODRS>
            <RECSRVRTID>387687138
            <RECURRINST>
               <NINSTS>36
               <FREQ>MONTHLY
            </RECURRINST>
            <PMTINFO>
               <BANKACCTFROM>
                  <BANKID>555432180
                  <ACCTID>763984
                  <ACCTTYPE>CHECKING
               </BANKACCTFROM>
               <TRNAMT>399.95                      <!-- changing amount -->
               <PAYEEID>77810
               <PAYACCT>444-78-97572
               <DTDUE>19971115
               <MEMO>Auto loan payment
            </PMTINFO>
            <MODPENDING>N
         </RECPMTMODRS>
      </RECPMTTRNRS>
   </BILLPAYMSGSRSV1>
</OFX>
```

12.12.7 Canceling a Recurring Payment

Cancel a recurring payment:

```
<OFX>
   <SIGNONMSGSRQV1>
      <SONRQ>
         <DTCLIENT>19971123101000
         <USERID>123-45-6789
         <USERPASS>MyPassword
         <LANGUAGE>ENG
         <FI>
            <ORG>NCH
            <FID>12321
         </FI>
         <APPID>MyApp
         <APPVER>0700
      </SONRQ>
   </SIGNONMSGSRQV1>
   <BILLPAYMSGSRQV1>
      <RECPMTTRNRQ>
         <TRNUID>10103
         <RECPMTCANCRQ>
            <RECSRVRTID>387687138
            <CANPENDING>Y
         </RECPMTCANCRQ>
      </RECPMTTRNRQ>
   </BILLPAYMSGSRQV1>
</OFX>
```

Server responds with:

```
<OFX>
   <SIGNONMSGSRSV1>
      <SONRS>
         <STATUS>
            <CODE>0
            <SEVERITY>INFO
         </STATUS>
         <DTSERVER>19971123101003
         <LANGUAGE>ENG
         <DTPROFUP>19961029101003
         <DTACCTUP>19961029101003
      </SONRS>
   </SIGNONMSGSRSV1>
   <BILLPAYMSGSRSV1>
      <RECPMTTRNRS>
         <TRNUID>10103
         <STATUS>
            <CODE>0
            <SEVERITY>INFO
         </STATUS>
         <RECPMTCANCRS>
            <RECSRVRTID>387687138
            <CANPENDING>Y
         </RECPMTCANCRS>
      </RECPMTTRNRS>
   </BILLPAYMSGSRSV1>
</OFX>
```

12.12.8 Adding a Payee to the Payee List

The user sends a request to add a payee to the user's payee list:

```
<OFX>
   <SIGNONMSGSRQV1>
      <SONRQ>
         <DTCLIENT>19970129101000
         <USERID>123-45-6789
         <USERPASS>MyPassword
         <LANGUAGE>ENG
         <FI>
            <ORG>NCH
            <FID>12321
         </FI>
         <APPID>MyApp
         <APPVER>0700
      </SONRQ>
   </SIGNONMSGSRQV1>
   <BILLPAYMSGSRQV1>
      <PAYEETRNRQ>
         <TRNUID>127688
         <PAYEERQ>
            <PAYEE>
               <NAME>ACME Rocket Works
               <ADDR1>101 Spring St.
               <ADDR2>Suite 503
               <CITY>Watkins Glen
               <STATE>NY
               <POSTALCODE>12345-6789
               <PHONE>888.555.1212
            </PAYEE>
            <PAYACCT>1001-99-8876
         </PAYEERQ>
      </PAYEETRNRQ>
   </BILLPAYMSGSRQV1>
</OFX>
```

The server responds:

```
<OFX>
   <SIGNONMSGSRSV1>
      <SONRS>
         <STATUS>
            <CODE>0
            <SEVERITY>INFO
         </STATUS>
         <DTSERVER>19961029101003
         <LANGUAGE>ENG
         <DTPROFUP>19961029101003
         <DTACCTUP>19961029101003
      </SONRS>
   </SIGNONMSGSRSV1>
   <BILLPAYMSGSRSV1>
      <PAYEETRNRS>
         <TRNUID>127688
         <STATUS>
            <CODE>0
            <SEVERITY>INFO
         </STATUS>
         <PAYEERS>
            <PAYEELSTID>78096786
            <PAYEE>
               <NAME>ACME Rocket Works
               <ADDR1>101 Spring St.
```

```
            <ADDR2>Suite 503
            <CITY>Watkins Glen
            <STATE>NY
            <POSTALCODE>12345-6789
            <PHONE>888.555.1212
         </PAYEE>
         <EXTDPAYEE>
            <PAYEEID>88878
            <IDSCOPE>GLOBAL
            <NAME>ACME Rocket Works, Inc.
            <DAYSTOPAY>2
         </EXTDPAYEE>
         <PAYACCT>1001-99-8876
      </PAYEERS>
   </PAYEETRNRS>
</BILLPAYMSGSRSV1>
</OFX>
```

12.12.9 Synchronizing Scheduled Payments

A client wishes to obtain all Payment and Recurring Payment transactions active on the server:

```
<OFX>
   <SIGNONMSGSRQV1>
      <SONRQ>
         <DTCLIENT>19971123101000
         <USERID>123-45-6789
         <USERPASS>MyPassword
         <LANGUAGE>ENG
         <FI>
            <ORG>NCH
            <FID>12321
         </FI>
         <APPID>MyApp
         <APPVER>0700
      </SONRQ>
   </SIGNONMSGSRQV1>
   <BILLPAYMSGSRQV1>
      <PMTSYNCRQ>
         <TOKEN>7647909384
         <REJECTIFMISSING>N
         <BANKACCTFROM>
            <BANKID>555432180
            <ACCTID>763984
            <ACCTTYPE>CHECKING
         </BANKACCTFROM>
      </PMTSYNCRQ>
      <RECPMTSYNCRQ>
         <TOKEN>9647409384
         <REJECTIFMISSING>N
         <BANKACCTFROM>
            <BANKID>555432180
            <ACCTID>763984
            <ACCTTYPE>CHECKING
         </BANKACCTFROM>
      </RECPMTSYNCRQ>
   </BILLPAYMSGSRQV1>
</OFX>
```

The server responds with one payment and one Recurring Payment, as well as current token values.

```
<OFX>
   <SIGNONMSGSRSV1>
      <SONRS>
```

```
      <STATUS>
         <CODE>0
         <SEVERITY>INFO
      </STATUS>
      <DTSERVER>19971123101003
      <LANGUAGE>ENG
      <DTPROFUP>19961029101003
      <DTACCTUP>19961029101003
   </SONRS>
</SIGNONMSGSRSV1>
<BILLPAYMSGSRSV1>
   <PMTSYNCRS>
      <TOKEN>3247989384
      <BANKACCTFROM>
         <BANKID>555432180
         <ACCTID>763984
         <ACCTTYPE>CHECKING
      </BANKACCTFROM>
      <PMTTRNRS>
         <TRNUID>10103
         <STATUS>
            <CODE>0
            <SEVERITY>INFO
         </STATUS>
         <PMTRS>
            <SRVRTID>1030155
            <PAYEELSTID>3486578
            <CURDEF>USD
            <PMTINFO>
               <BANKACCTFROM>
                  <BANKID>123432123
                  <ACCTID>516273
                  <ACCTTYPE>CHECKING
               </BANKACCTFROM>
               <TRNAMT>123.45
               <PAYEE>
                  <NAME>J. C. Counts
                  <ADDR1>100 Main St.
                  <CITY>Turlock
                  <STATE>CA
                  <POSTALCODE>90101
                  <PHONE>415.987.6543
               </PAYEE>
               <PAYACCT>10101
               <DTDUE>19971001
               <MEMO>payment #3
            </PMTINFO>
            <EXTDPAYEE>
               <PAYEEID>9076
               <IDSCOPE>USER
               <NAME>J. C. Counts
               <DAYSTOPAY>3
            </EXTDPAYEE>
            <PMTPRCSTS>
              <PMTPRCCODE>WILLPROCESSON
              <DTPMTPRC>19971001
            </PMTPRCSTS>
         </PMTRS>
      </PMTTRNRS>
   </PMTSYNCRS>
   <RECPMTSYNCRS>
      <TOKEN>94879835
      <BANKACCTFROM>
         <BANKID>555432180
```

```
            <ACCTID>763984
            <ACCTTYPE>CHECKING
         </BANKACCTFROM>
         <RECPMTTRNRS>
            <TRNUID>10202
            <STATUS>
               <CODE>0
               <SEVERITY>INFO
            </STATUS>
            <RECPMTRS>
               <RECSRVRTID>387687138
               <PAYEELSTID>938469
               <CURDEF>USD
               <RECURRINST>
                  <NINSTS>36
                  <FREQ>MONTHLY
               </RECURRINST>
               <PMTINFO>
                  <BANKACCTFROM>
                     <BANKID>555432180
                     <ACCTID>763984
                     <ACCTTYPE>CHECKING
                  </BANKACCTFROM>
                  <TRNAMT>395.00
                  <PAYEEID>77810
                  <PAYACCT>444-78-97572
                  <DTDUE>19971115
                  <MEMO>Auto loan payment
               </PMTINFO>
            </RECPMTRS>
         </RECPMTTRNRS>
      </RECPMTSYNCRS>
   </BILLPAYMSGSRSV1>
</OFX>
```

13. Investments

Open Financial Exchange supports download of security information and detailed investment account statements including transactions, open orders, balances, and positions.

Client Sends	*Server Responds*
Account identifier	
Whether to download open orders	
Whether to download transactions	
Date range if transactions should be downloaded	
Whether to download positions	
Whether to download balances	
Additional securities to send information about	
	Date and time for statement
	Default currency for statement
	Account identifier
	Investment transactions
	Banking transactions
	Open orders
	Positions
	Account balances
	Available Cash Balance
	Short Balance
	Margin Balance
	Buying power
	Marketing message
	List of securities

Note: *This release of Open Financial Exchange does not support trading or tax lots.*

13.1 Types of Response Information

The response consists of five types of information:

- Transactions – a combination of bank transaction detail records and investment transaction detail records. Transactions only within the specified start and stop dates are sent.
- Positions – positions a user has at a brokerage. Each statement response must contain a complete set of position records, even if no transactions occurred in the requested statement period for a particular holding.
- Balances – current balances typically reported on an FI statement, such as cash balance or buying power. They can also convey other numbers of interest, such as current interest rates.
- Open Orders – current open trading orders that a user has at a brokerage.
- Securities – any security referenced in either transactions, positions, open orders or explicitly requested.

13.2 Sub-Accounts

Many FIs distinguish between activity and positions in cash, margin, and short accounts, with some FIs having many other types of "sub-accounts." Open Financial Exchange defines four standard types of sub-accounts: Cash, Margin, Short, Other. Position, Transaction, and Open Order records identify the sub-account.

13.3 Units, Precision, and Signs

This section provides information about numerical values for investment transactions. For more information about common data types used within Open Financial Exchange, refer to Chapter 3, "Common Aggregates, Elements, and Data Types."

13.3.1 Units

The units for security units and unit price are those commonly used on brokerage statements, and differ for each type of security.

- Stocks and Other – use number of shares for units and dollar value for unit price.
- Mutual Funds – in most cases shares are used, but in some cases the dollar value is used. The unit type is specified in cases in which it can be either.
- Bonds – use face value for units and percentage of par for unit price. For example, a $25,000 bond trading at $88 would use 25000 as the units and 88 as the unit price.
- Options – use number of contracts (not shares) for units, and price per share (not contract) for unit price.

13.3.2 Precision

Open Financial Exchange does not specify the precision of fields since the precision is client-dependent. However, it is recommended that clients and servers follow these rules:

- Clients and servers should send as much precision as they have
- Clients and servers should use a precision equal to or better than 1/256 of a share

13.3.3 Signs

Chapter 3, “Common Aggregates, Elements, and Data Types,” describes how to use positive and negative numbers. Briefly, quantities and total values should be signed from the perspective of the user. In a stock buy, the total value is negative, the unit price is always positive, and the number of units is positive.

UNITS and TOTALS are signed from the perspective of the user (positive currency amount for SELLs, negative currency amount for BUYs). All other Investment transaction amounts are always positively signed. In other words UNITPRICE, COMMISSION, FEES, TAXES, WITHHOLDING, LOAD, MARKUP and MARKDOWN are always positive numbers.

A positive COMMISSION, TAXES, LOAD, WITHHOLDING, or FEE increased the negatively signed TOTAL on a BUYSTOCK and decreased the positively signed TOTAL on a SELLSTOCK. MARKUP and MARKDOWN increase or decrease, respectively, the UNITPRICE.

Servers should return corrections to investment buys or sells as the opposite transaction type, e.g., a correction to a buy is returned as a sell. A correction to MARGININTEREST or RETOFCAP is returned as an INVEXPENSE.

13.4 Bank and Investment Transactions

Many FIs provide investment accounts that allow users to write checks and perform other traditional banking transactions, as well as investment transactions. Open Financial Exchange requires FIs to indicate in the download whether check-writing privileges exist for a given account.

FIs need to use the correct transaction record, bank or investment, for each real-world transaction. Use the following guidelines:

- Checks, electronic funds transfers, and ATM transactions associated with CMA or money market sweep accounts are always represented with a bank transaction record.
- Investment actions that involve securities (buy, sell, stock split, reinvest, etc.) are always represented with an investment record. Actions that are cash-only but are directly associated with a security are also investment actions (for example, dividends).
- Other cash-only actions require careful analysis by the FI. Those that affect investment performance analysis should be sent using the appropriate investment action (investment income - miscellaneous, investment expense). Those that are completely unrelated to investment should be sent as a bank record.

13.5 Money Market Funds

Money market funds can be handled in one of three different ways depending on how the fund is modeled at the financial institution

- Separate account at the financial institution
- Sweep account within an investment account
- Position within an investment account

13.5.1 Separate Account at the Financial Institution

In this case, the money market fund is in its own account with its own account number, distinct from the investment account. In Open Financial Exchange, you should model the money market fund as a separate money market bank account; see Chapter 11, "Banking." The banking <STMTRQ> request aggregate and <STMTRS> response aggregate will be used to download transactions.

13.5.2 Sweep Account Within an Investment Account

Open Financial Exchange uses the money market as a "sweep" account, where cash is "swept" as needed when buying and selling securities. The money market fund does not have its own account number. The customer sees the money market fund as an investment-account cash balance. In Open Financial Exchange, checks, ATMs, electronic fund transfer, deposit, and withdrawal transactions should be downloaded using banking transactions within the investment account. However, the sweep transactions in and out of the money market fund should not be downloaded to the client.

13.5.3 Position Within an Investment Account

The customer purchases the money market fund and is held in the account as a position. The money market fund does not have its own account number. In Open Financial Exchange, the money market fund should be returned as a <POSOTHER> position in the <INVPOSLIST>, with a <UNITPRICE> of 1.00 and <UNITS> as the current value of the position. Purchases and redemptions should be modeled as <BUYOTHER> and <SELLOTHER> transactions with a <UNITPRICE> of 1.00 and <UNITS> as the transaction amount.

13.6 Investment Accounts

Investment account information is downloaded using the account information response aggregate <ACCTINFORS>. For more information, refer to Chapter 8, "Activation & Account Information." <INVACCTFROM> specifies the account. The <INVACCTINFO> aggregate specifies the investment-specific information.

13.6.1 Specifying the Investment Account <INVACCTFROM>

Tag	*Description*
<INVACCTFROM>	Account-from aggregate
<BROKERID>	Unique identifier for the FI, ***A-22***
<ACCTID>	Account number at FI, ***A-22***
</INVACCTFROM>	

Brokers should use the domain name of their company's URL as the BROKERID, e.g.,

If URL=www.broker.com
then BROKERID=broker.com

The <INVACCTTO> aggregate contains the same elements.

13.6.2 Investment Account Information <INVACCTINFO>

The <INVACCTINFO> aggregate should appear in the <ACCTINFO> aggregate for accounts that support investment statement download. For more information about the <ACCTINFO> aggregate, refer to Chapter 8, "Activation & Account Information."

Tag	*Description*
<INVACCTINFO>	Investment-account-information-record aggregate
<INVACCTFROM>	Account at FI
</INVACCTFROM>	
<USPRODUCTTYPE>	Classification of account. See section 13.6.2.1 for values
<CHECKING>	Whether the account has check writing privileges, ***Boolean***
<SVCSTATUS>	Activation status for investment statement download for the account. ACTIVE (signed up), PEND (in the process of signing up), AVAIL (have not signed up).
<INVACCTTYPE>	Type of account. INDIVIDUAL, JOINT, TRUST, CORPORATE
<OPTIONLEVEL>	Text description of option trading privileges, ***A-40***
</INVACCTINFO>	

If an investment account has payments functionality, the analogous PMTINFO aggregate (see 12.5.1) should also be sent in the ACCTINFO for the account. Payment information will be sent using the message sets described in the Chapter 12, "Payments."

13.6.2.1 Values for <USPRODUCTTYPE>

<USPRODUCTTYPE> classifies accounts according to their account type. Valid values are:

Product Type	*Description*
401K	A 401(K) account
403B	A 403(B) account
IRA	An IRA account
KEOGH	Keogh (Money Purchase/Profit Sharing)
OTHER	Other account type
SARSEP	Salary Reduction Simplified Employer Pension plan
SIMPLE	Savings Incentive Match Plan for employees
NORMAL	Regular account
TDA	Tax Deferred Annuity
TRUST	Trust (including UTMA)
UGMA	Custodial account

13.6.2.2 International Note

The <USPRODUCTTYPE> tag is intended for use by FIs in the United States. Open Financial Exchange will be expanded to provide equivalent tags to support the needs for other countries.

13.6.3 Brokerage, Mutual Fund, and 401K Accounts

Investment accounts include brokerage accounts, mutual fund accounts, 401K accounts, and other retirement accounts. Open Financial Exchange supports transactions, positions, balances, and open orders for all of these account types.

13.7 Investment Message Sets and Profile

Open Financial Exchange separates messages that the client and server send into groups called message sets. Each financial institution defines the message sets that the institution supports. The messages described in this chapter fall into two message sets:

- Investment Statement Download
- Security Information

Each message set contains options that allow a financial institution to customize its use of Open Financial Exchange. For example, an institution can support the Investment Statement Download Set (INVSTMTMSGSETV1 and/or INVSTMTMSGSETV2), but it can choose not to support the download of open orders.

The options and attributes are defined in the profile as part of each message set definition. Each set of options and attributes appears within an aggregate that is specific to a message set. For example, <INVSTMTMSGSETV1> and/or <INVSTMTMSGSETV2> contains all the options and attributes that pertain to investment statement download.

13.7.1 Investment Statement Download

13.7.1.1 Investment Statement Message Set Profile <INVSTMTMSGSET>

The investment statement message set profile aggregate <INVSTMTMSGSET> is used in the response to a financial institution profile request (see Chapter 7, “FI Profile”) to specify which activities it supports.

Tag	*Description*
<INVSTMTMSGSET>	Investment-statement-message-set-profile aggregate
<INVSTMTMSGSETV1>	Version 1 message set
<MSGSETCORE>	Common message set information, see Chapter 7, “FI Profile”
</MSGSETCORE>	
<TRANDNLD>	Whether the FI server downloads investment statement transactions, ***Boolean***
<OODNLD>	Whether the FI server downloads investment open orders, ***Boolean***
<POSDNLD>	Whether the FI server downloads investment statement positions, ***Boolean***
<BALDNLD>	Whether the FI server downloads investment balances, ***Boolean***
<CANEMAIL>	Whether the FI supports investment e-mail. Use generic e-mail profile to specify whether generic e-mail is supported; see Chapter 9, “Customer to FI Communication.” ***Boolean***
</INVSTMTMSGSETV1>	
<INVSTMTMSGSETV2>	Version 2 message set
<MSGSETCORE>	Common message set information, see Chapter 7, “FI Profile”

Tag	Description
</MSGSETCORE>	
<TRANDNLD>	Whether the FI server downloads investment statement transactions, ***Boolean***
<OODNLD>	Whether the FI server downloads investment open orders, ***Boolean***
<POSDNLD>	Whether the FI server downloads investment statement positions, ***Boolean***
<BALDNLD>	Whether the FI server downloads investment balances, ***Boolean***
<CANEMAIL>	Whether the FI supports investment e-mail. Use generic e-mail profile to specify whether generic e-mail is supported; see Chapter 9, "Customer to FI Communication." ***Boolean***
</INVSTMTMSGSETV2>	
</INVSTMTMSGSET>	

13.7.1.2 Investment Statement Message Set and Messages

13.7.1.2.1 Investment Statement Message Set Request Messages

Tag	Description
<INVSTMTMSGSET>	
<INVSTMTMSGSETV1>	
<INVSTMTMSGSRQV1>	INVSTMTTRNRQ INVSTMTTRQ INVMAILTRNRQ INVMAILRQ INVMAILSYNCRQ
</INVSTMTMSGSRQV1>	
</INVSTMTMSGSETV1>	
<INVSTMTMSGSETV2>	
<INVSTMTMSGSRQV2>	INVSTMTTRNRQ INVSTMTTRQ INVMAILTRNRQ INVMAILRQ INVMAILSYNCRQ
</INVSTMTMSGSRQV2>	
</INVSTMTMSGSETV2>	
</INVSTMTMSGSET>	

13.7.1.2.2 Investment Statement Message Set Response Messages

Tag	Description
<INVSTMTMSGSET>	
<INVSTMTMSGSETV1>	
<INVSTMTMSGSRSV1>	INVSTMTTRNRS INVSTMTRS

Tag	Description
	INVMAILTRNRS
	INVMAILRS
	INVMAILSYNCRS
</INVSTMTMSGSRSV1>	
</INVSTMTMSGSETV1>	
<INVSTMTMSGSETV2>	
<INVSTMTMSGSRSV2>	INVSTMTTRNRS
	INVSTMTRS
	INVMAILTRNRS
	INVMAILRS
	INVMAILSYNCRS
</INVSTMTMSGSRSV2>	
</INVSTMTMSGSETV2>	
</INVSTMTMSGSET>	

13.7.2 Security Information

13.7.2.1 Security List Message Set Profile <SECLISTMSGSET>

The security list message set profile aggregate <SECLISTMSGSET> is used in the response to an FI profile request (see Chapter 7, “FI Profile”) to specify which activities it supports.

Tag	Description
<SECLISTMSGSET>	Security-information-message-set-profile aggregate
<SECLISTMSGSETV1>	Version 1 message set
<MSGSETCORE>	Common message set information, see Chapter 7, “FI Profile”
</MSGSETCORE>	
<SECLISTRQDNLD>	Whether the FI server responds to security list requests, ***Boolean***
</SECLISTMSGSETV1>	
<SECLISTMSGSETV2>	Version 2 message set
<MSGSETCORE>	Common message set information, Chapter 7, “FI Profile”
</MSGSETCORE>	
<SECLISTRQDNLD>	Whether the FI server responds to security list requests, ***Boolean***
</SECLISTMSGSETV2>	
</SECLISTMSGSET>	

13.7.2.2 Security List Message Set and Messages

13.7.2.2.1 Security List Message Set Request Messages

Tag	Description
<SECLISTMSGSET>	
<SECLISTMSGSETV1>	
<SECLISTMSGSRQV1>	SECLISTTRNRQ

Tag	Description
	SECLISTRQ
</SECLISTMSGSRQV1>	
</SECLISTMSGSETV1>	
<SECLISTMSGSETV2>	
<SECLISTMSGSRQV2>	SECLISTTRNRQ
	SECLISTRQ
</SECLISTMSGSRQV2>	
</SECLISTMSGSETV2>	
</SECLISTMSGSET>	

13.7.2.2.2 Security List Message Set Response Messages

Tag	Description
<SECLISTMSGSET>	
<SECLISTMSGSETV1>	
<SECLISTMSGSRSV1>	SECLISTTRNRS
	SECLISTRS
</SECLISTMSGSRSV1>	
</SECLISTMSGSETV1>	
<SECLISTMSGSETV2>	
<SECLISTMSGSRSV2>	SECLISTTRNRS
	SECLISTRS
</SECLISTMSGSRSV2>	
</SECLISTMSGSETV2>	
</SECLISTMSGSET>	

13.8 Investment Securities

13.8.1 Security Identification <SECID>

Securities must be consistently identified to allow client applications to prepare accurate investment reports across all user investment accounts, even at multiple FIs. At this time, neither a security name nor its symbol is standardized. Therefore, Open Financial Exchange uses CUSIP numbers (a unique 9-digit alphanumeric identifier) to identify securities. CUSIP numbers are available for the vast majority of securities traded today, including those without symbols such as bonds. For a security that does not have a CUSIP, a financial institution must follow the standard procedure of assigning a CUSIP by using itself as the issuer to avoid conflict with any other CUSIP.

Tag	Description
<SECID>	Security-identifier aggregate
<UNIQUEID>	Unique identifier for the security. CUSIP for US FIs. ***A-32***
<UNIQUEIDTYPE>	Name of standard used to identify the security i.e., “CUSIP” for FIs in the United States, ***A-10***
</SECID>	

13.8.1.1 International Note

Non-US financial institutions that do not have access to CUSIP numbers must supply a unique identifier for each security in the UNIQUEID field of this aggregate. Open Financial Exchange will be expanded to include other security identifying standards.

13.8.2 Security List Request

The user can use the SECLISTTRNRQ and SECLISTRQ aggregates to request information about specific securities. The SECLISTTRNRQ is the transaction-level aggregate that contains the SECLISTRQ. The SECLISTRQ aggregate specifies for which securities information is being requested.

13.8.2.1 Security List Transaction Request <SECLISTTRNRQ>

Tag	*Description*
<SECLISTTRNRQ>	Transaction-request aggregate
<TRNUID>	Client-assigned globally unique ID for this transaction ***trnuid***
<CLTCOOKIE>	Data to be echoed in the transaction response, ***A-32***
<TAN>	Transaction authorization number; used in some countries with some types of transactions. Country-specific documentation will define messages that require a <TAN>, ***A-80***
<SECLISTRQ>	Aggregate for the security list request (see section 13.8.2.2)
</SECLISTRQ>	
</SECLISTTRNRQ>	

13.8.2.2 Security List Request <SECLISTRQ>

For the security list request, securities must be specified with either a SECID aggregate, a ticker symbol, or an FI assigned identifier.

Tag	*Description*
<SECLISTRQ>	Security-list-request aggregate
<SECRQ>	Security request (one or more)
Security identification. Specify either <SECID>, <TICKER>, or <FIID>.	
<SECID>	Security identifier aggregate
</SECID>	
-or-	
<TICKER>	Ticker symbol, ***A-32***
-or-	
<FIID>	FI specific ID for the security, ***A-32***
</SECRQ>	
</SECLISTRQ>	

13.8.3 Security List Response

If the client sends a security list request to an FI, then the server must send back a security list response to the client application. The security list response is used primarily to report the status of the security list request. The actual security information should be sent in the security list SECLIST aggregate described in section 13.8.4.

13.8.3.1 Security List Transaction Response <SECLISTTRNRS>

Tag	*Description*
<SECLISTTRNRS>	Transaction-response aggregate
<TRNUID>	Client-assigned globally unique ID for this transaction ***trnuid***
<STATUS>	Status aggregate
</STATUS>	
<CLTCOOKIE>	Client-provided data, **REQUIRED** if provided in request, ***A-32***
<SECLISTRS>	Aggregate for the security list response, see 0
</SECLISTRS>	
</SECLISTTRNRS>	

13.8.3.2 Status Codes

Code	*Meaning*
0	Success
2000	General error (ERROR)
2019	Duplicate request (ERROR)
12500	One or more securities not found (ERROR)

13.8.3.3 Security List Response <SECLISTRS>

The security list response aggregate, the only empty aggregate in OFX, is used to respond to the <SECLISTRQ>. It is used to signify that the security list is generated as a result of a security list request. The actual security information should be included in the <SECLIST> aggregate.

Tag	*Description*
<SECLISTRS>	Security-list-response (the only empty aggregate in OFX).
</SECLISTRS>	

13.8.4 Security List <SECLIST>

The SECLIST should be sent in the following two cases.

- In response to a SECLISTTRNRQ sent by a client application where the SECLIST should contain information for each security specified in the SECLISTTRNRQ.
- When the response file contains an investment statement download that has positions, transactions, or open orders. The SECLIST should contain information about security referenced in the investment statement download. Clients are completely dependent on the security list to provide descriptive information for the securities referenced in positions, transactions, and open orders.

Tag	*Description*
<SECLIST>	Security-list-request aggregate
<***XXX***INFO>	Security information aggregates (one or more): <DEBTINFO>, <MFINFO>, <OPTINFO>, <OTHERINFO>, or <STOCKINFO>.

Tag	Description
</***XXX***INFO>	
</SECLIST>	

13.8.5 Securities Information

The <MFINFO>, <STOCKINFO>, <OPTINFO>, <DEBTINFO>, and <OTHERINFO> aggregates provide security information. They define the type of security, and one or more sets of descriptive information. These aggregates relate the <SECID> used in positions, transactions, and open orders to descriptive information about those securities. In this way, the system describes a given security only once, no matter how many times it is referenced.

13.8.5.1 General Securities Information <SECINFO>

The <SECINFO> aggregate contains fields that are common to all security types. This aggregate is used in the security type specific aggregates in the following sections.

Tag	Version	Description
<SECINFO>		Security-information aggregate
<SECID>		Security-identifier aggregate
</SECID>		
<SECNAME>		Full name of security, ***A-120***
<TICKER>		Ticker symbol (at most one), ***A-32***
<FIID>		FI ID number for this security (at most one), ***A-32***
<RATING>		Rating, ***A-10***
<UNITPRICE>		Current price of security, ***unitprice***
<DTASOF>		Date as of for the unit price, ***datetime***
<CURRENCY>		Overriding currency aggregate for unit price, see section 5.2
</CURRENCY>		
<MEMO>	V1 only	***Memo***
<MEMO2>	V2 only	***Memo2***
</SECINFO>		

13.8.5.2 Debt Information <DEBTINFO>

Tag	Description
<DEBTINFO>	Opening tag for debt information aggregate
<SECINFO>	Security information aggregate
</SECINFO>	
<PARVALUE>	Par value, ***amount***
<DEBTTYPE>	Debt type (at most one) COUPON = coupon ZERO = zero coupon
<DEBTCLASS>	Classification of debt. TREASURY, MUNICIPAL, CORPORATE, OTHER.
<COUPONRT>	Bond coupon rate for next closest call date (at most one), ***rate***
<DTCOUPON>	Maturity date for next coupon, ***date***
<COUPONFREQ>	When coupons mature. One of the following values: MONTHLY, QUARTERLY,

Tag	Description
	SEMIANNUAL, ANNUAL, or OTHER.
<CALLPRICE>	Bond call price (at most one), ***unitprice***
<YIELDTOCALL>	Yield to next call, ***rate***
<DTCALL>	Next call date (at most one), ***date***
<CALLTYPE>	Type of next call. CALL, PUT, PREFUND, MATURITY
<YIELDTOMAT>	Yield to maturity, ***rate***
<DTMAT>	Debt maturity date (at most one), ***date***
<ASSETCLASS>	Asset Class (at most one), DOMESTICBOND, INTLBOND, LARGESTOCK SMALLSTOCK, INTLSTOCK, MONEYMRKT, OTHER
<FIASSETCLASS>	Text string containing an FI defined asset class, ***A-32***
</DEBTINFO>	

13.8.5.3 Mutual Fund Information <MFINFO>

Tag	Description
<MFINFO>	Mutual-fund-information aggregate
<SECINFO>	Security-information aggregate
</SECINFO>	
<MFTYPE>	Mutual fund type. OPENEND, CLOSEEND, OTHER
<YIELD>	Current yield reported as portion of the fund's assets (at most one), ***rate***
<DTYIELDASOF>	As-of date for yield value, ***datetime***
<MFASSETCLASS>	Asset class breakdown for the mutual fund
<PORTION>	Portion of the mutual fund with a specific asset classification (one or more)
<ASSETCLASS>	Asset Class, DOMESTICBOND, INTLBOND, LARGESTOCK SMALLSTOCK, INTLSTOCK, MONEYMRKT, OTHER
<PERCENT>	Percentage of the fund that falls under this asset class, ***rate***
</PORTION>	
</MFASSETCLASS>	
<FIMFASSETCLASS>	FI defined asset class breakdown for the mutual fund
<FIPORTION>	Portion of the mutual fund with a specific asset classification (one or more)
<FIASSETCLASS>	Text string containing an FI defined asset class, ***A-32***
<PERCENT>	Percentage of the fund that falls under this asset class, ***rate***
</FIPORTION>	
</FIMFASSETCLASS>	
</MFINFO>	

13.8.5.4 Option Information <OPTINFO>

Tag	Description
<OPTINFO>	Option-information aggregate
<SECINFO>	Security-information aggregate

Tag	Description
</SECINFO>	
<OPTTYPE>	Option type: PUT = put CALL = call
<STRIKEPRICE>	Strike price ***unitprice***
<DTEXPIRE>	Expiration date, ***date***
<SHPERCTRCT>	Shares per contract, ***N-5***
<SECID>	Security ID of the underlying security
</SECID>	
<ASSETCLASS>	Asset Class (at most one), DOMESTICBOND, INTLBOND, LARGESTOCK SMALLSTOCK, INTLSTOCK, MONEYMRKT, OTHER
<FIASSETCLASS>	Text string containing an FI defined asset class, ***A-32***
</OPTINFO>	

13.8.5.5 Other Security Type Information <OTHERINFO>

Use this aggregate for security types other than debts, mutual funds, options, and stocks.

Tag	Description
<OTHERINFO>	Other aggregate.
<SECINFO>	Security information aggregate
</SECINFO>	
<TYPEDESC>	Description of security type, ***A-32***
<ASSETCLASS>	Asset Class (at most one), DOMESTICBOND, INTLBOND, LARGESTOCK SMALLSTOCK, INTLSTOCK, MONEYMRKT, OTHER
<FIASSETCLASS>	Text string containing an FI defined asset class, ***A-32***
</OTHERINFO>	

13.8.5.6 Stock Information <STOCKINFO>

Tag	Description
<STOCKINFO>	Stock-information aggregate
<SECINFO>	Security-information aggregate
</SECINFO>	
<STOCKTYPE>	Stock type: COMMON, PREFERRED, CONVERTIBLE, OTHER
<YIELD>	Current yield reported as the dividend expressed as a portion of the current stock price (at most one), ***rate***
<DTYIELDASOF>	As-of date for yield value, ***datetime***
<ASSETCLASS>	Asset Class (at most one): DOMESTICBOND, INTLBOND, LARGESTOCK SMALLSTOCK, INTLSTOCK, MONEYMRKT, OTHER
<FIASSETCLASS>	Text string containing an FI defined asset class, ***A-32***
</STOCKINFO>	

13.8.5.7 Asset Class Descriptions

Asset Class	Description

Asset Class	Description
DOMESTICBOND	The Domestic Bonds asset class consists of government or corporate bonds issued in the United States.
INTLBOND	The International Bonds asset class consists of government or corporate bonds issued in foreign countries or the United States.
LARGESTOCK	The Large Cap Stocks asset class consists of stocks for U.S. companies with market capitalizations of $2 billion or more.
SMALLSTOCK	The Small Cap Stocks asset class consists of stocks for U.S. companies with market capitalizations of approximately $100 million to $2 billion.
INTLSTOCK	The International Stocks asset class consists of publicly-traded stocks for companies based in foreign countries.
MONEYMRKT	The Money Market asset class consists of stable, short-term investments which provide income that rises and falls with short-term interest rates.
OTHER	The Other asset class consists of investments which do not fit in any of the other asset classes.

13.9 Investment Statement Download

Investment statement download allows a customer to receive transactions, positions, open orders, and balances that are typically part of a regular paper statement. Clients can retrieve statement data on a daily basis if they wish.

Clients usually allow customers to view investment transactions and guide customers through a process of updating their account registers based on the downloaded transactions. By using transaction IDs supplied by FIs, Open Financial Exchange makes it possible for clients to insure that each transaction is downloaded only once. The request also contains starting and ending dates to limit the amount of downloaded data. Clients can remember the last date they receive a download, and use that date as the starting date in the next request.

Investment statement download requires the client to designate an account for the download, and to indicate what type of data should be downloaded. If the client wishes to download transactions, it can specify a date range that the transactions fall within. The server returns transactions that match the date range, if one is specified. If a date range is not specified, the server returns all available transactions for the account.

13.9.1 Investment Statement Request

Investment statement download can be requested using the INVSTMTTRNRQ and INVSTMTRQ aggregates. The INVSTMTTRNRQ is the transaction level aggregate that contains the INVSTMTRQ. The INVSTMTRQ aggregate specifies what types of information to include in the statement download and from which account to download the information.

13.9.1.1 Investment Statement Transaction Request <INVSTMTTRNRQ>

Tag	Description
<INVSTMTTRNRQ>	Transaction-request aggregate
<TRNUID>	Client-assigned globally unique ID for this transaction, ***trnuid***
<CLTCOOKIE>	Data to be echoed in the transaction response, ***A-32***
<TAN>	Transaction authorization number; used in some countries with some types of transactions. Country-specific documentation will define messages that require a <TAN>, ***A-80***
<INVSTMTRQ>	Aggregate for the investment statement download request (see section

Tag	Description
	13.9.1.2)
</INVSTMTRQ>	
</INVSTMTTRNRQ>	

13.9.1.2 Investment Statement Request <INVSTMTRQ>

The following table shows the Investment Statement Request record. It is similar to a bank statement request, except that there are extra tags to indicate which pieces the user desires. Note that because transaction and position requests require date information, they use aggregates, whereas the other requests are elemental of type Boolean.

Clients and servers should interpret <DTSTART> and <DTEND> as described in Chapter 3, "Common Aggregates, Elements, and Data Types."

If <DTASOF> is not included with the <INCPOS> aggregate, the server should return the most current position information available.

Tag	Description and Type
<INVSTMTRQ>	Investment-request aggregate
<INVACCTFROM>	Account-from aggregate
</INVACCTFROM>	
<INCTRAN>	Include-transactions aggregate (at most one)
<DTSTART>	Start date of request, ***datetime***
<DTEND>	Ending date of request (at most one),.***datetime***
<INCLUDE>	Whether to include transactions in the statement download, ***Boolean***
</INCTRAN>	
<INCOO>	Include investment open orders in response, ***Boolean***
<INCPOS>	Include investment positions in response
<DTASOF>	Date that positions should be sent down for, ***datetime***
<INCLUDE>	Whether to include positions in the statement download, ***Boolean***
</INCPOS>	
<**INCBAL** >	Include investment balance in response, ***Boolean***
</**INVSTMTRQ**>	

13.9.2 Investment Statement Response

13.9.2.1 Investment Statement Transaction Response <INVSTMTTRNRS>

Tag	Description
<INVSTMTTRNRS>	Transaction-response aggregate
<TRNUID>	Client-assigned globally unique ID for this transaction, ***trnuid***
<STATUS>	Status aggregate
</STATUS>	
<CLTCOOKIE>	Client-provided data, **REQUIRED** if provided in request, ***A-32***
<INVSTMTRS>	Aggregate for the investment statement download response (see section 13.9.2.2)

Tag	Description
</INVSTMTRS>	
</INVSTMTTRNRS>	

13.9.2.2 Investment Statement Response <INVSTMTRS>

The response can contain transaction, position, open order, and/or balance detail records; each in its own aggregate. The transaction list aggregate can contain a mixture of bank statement records and investment transactions, as specified below.

Tag	Description
<INVSTMTRS>	Investment-response aggregate
<DTASOF>	As of date & time for the statement download, ***datetime***
<CURDEF>	Default currency for the statement, ***currsymbol***
<INVACCTFROM>	Which account at FI
</INVACCTFROM>	
<INVTRANLIST>	Begin transaction list (at most one)
<DTSTART>	Start date for transaction data, ***datetime***
<DTEND>	This is the value that should be sent in the next <DTSTART> request to insure that no transactions are missed, ***datetime***
(investment transaction aggregates)	Investment statement transaction aggregates (zero or more); see section 13.9.2.4.4.
<INVBANKTRAN>	Banking-related transactions for the investment account (zero or more)
</INVBANKTRAN>	(See section 13.9.2.3)
</INVTRANLIST>	End of investment transaction list
<INVPOSLIST>	Beginning of investment position list (at most one)
<POSxxxxx>	Security type specific position aggregates (zero or more): POSMF, POSSTOCK, POSDEBT, POSOPT, POSOTHER
</POSxxxxx>	
</INVPOSLIST>	End of investment position list
<INVBAL>	Balances aggregate, see section 13.9.2.7
</INVBAL>	
<INVOOLIST>	Beginning of investment open order list (at most one)
<OOxxxxx>	Action and security type specific open order aggregates (zero or more): see section 13.9.2.5.2
</OOxxxxx>	
</INVOOLIST>	End of investment open order list
<MKTGINFO>	Marketing information (at most one), ***A-360***.
</INVSTMTRS>	

The various sections of the investment statement download are returned only if requested.

13.9.2.2.1 Note on Margin Calls

For investment statement download, margin call information should be included in the balances section. Margin call information should be contained in a <BAL> aggregate and included in the balance list <BALLIST>.

13.9.2.3 Bank Transactions <INVBANKTRAN>

Use the INVBANKTRAN aggregate to download bank transactions in an investment statement download.

Tag	*Description*
<INVBANKTRAN>	Banking related transactions for the investment account
<STMTTRN>	Bank (cash) transaction aggregates
</STMTTRN>	(See Chapter 11, "Banking")
<SUBACCTFUND>	The sub-account associated with the funds for the transaction; see section 13.9.2.4.2
</INVBANKTRAN>	

13.9.2.4 Investment Transactions

Note that the following types of investment actions found on statements should ***not*** be sent in Open Financial Exchange:

- Transaction-specific miscellaneous fees – fees that affect the basis of the transaction should be incorporated into the <COMMISSION>, <FEES>, <LOAD>, or <TAXES> amounts.
- Settlement actions.
- Sweeps, unless handled as any other investment position.

For transactions that involve securities, the client can create transactions based on the formula

total = (units * (unitprice +/- markup/markdown)) +/- (commission + fees + load + taxes)
(after adjusting quantity and unitprice to standard units based on the type of security.)

Thus, it is important the FIs incorporate all other transactional fees into the commission field. Clients can account for bond accrued interest and withholdings using separate client transactions.

13.9.2.4.1 General Transaction Aggregate <INVTRAN>

The INVTRAN aggregate contains fields common to many of the investment transactions. It is referenced within the transaction aggregates in the following sections.

Each <INVTRAN> contains an <FITID> that the client uses to detect whether the server previously downloaded the transaction.

Tag		*Description*
<INVTRAN>		Investment-transaction-response aggregate
<FITID>		Unique FI-assigned transaction ID. This ID is used to detect duplicate downloads. ***FITID***
<SRVRTID>	V1 only	Server assigned transaction ID, ***SRVRTID***
<SRVRTID2>	V2 only	Server assigned transaction ID, ***SRVRTID2***
<DTTRADE>		Trade date; for stock splits, day of record, ***datetime***
<DTSETTLE>		Settlement date; for stock splits, execution date, ***datetime***
<MEMO>	V1 only	Other information about transaction (at most one), ***memo***
<MEMO2>	V2 only	Other information about transaction (at most one), ***memo2***
</INVTRAN>		

13.9.2.4.2 Transaction Aggregate Elements

The following tags are referenced within of the following investment transaction aggregates.

Tag	*Version*	*Description*
<ACCRDINT>		For debt purchases, accrued interest, ***amount***
<AVGCOSTBASIS>		Average cost basis, ***amount***
<BUYTYPE>		Type of purchase: BUY, BUYTOCOVER
<COMMISSION>		Transaction commission. ***Amount***
<DENOMINATOR>	V2 change	For stock splits, split ratio denominator, ***Integer***
<DTPURCHASE>		The security's original purchase date, ***date***
<GAIN>		For sales, total gain, ***amount***
<FEES>		Fees applied to trade, ***amount***
<FRACCASH>		Cash for fractional units., (used for stock splits), ***amount***
<INCOMETYPE>		Type of investment income: CGLONG (capital gains-long term), CGSHORT (capital gains-short term), DIV (dividend), INTEREST, MISC
<LOAD>		Load on the transaction, ***amount***
<MARKDOWN>		Portion of the unit price that is attributed to the dealer markdown, ***unitprice***
<MARKUP>		Portion of the unit price that is attributed to the dealer markup, ***unitprice***
<NEWUNITS>		For stock splits, number of shares after the split, ***quantity***
<NUMERATOR>	V2 change	For stock splits, split ratio numerator, ***Integer***
<OLDUNITS>		For stock splits, number of shares before the split, ***quantity***
<OPTACTION>		For options, action type: EXERCISE, ASSIGN, EXPIRE
<OPTBUYTYPE>		For options, type of purchase: BUYTOOPEN, BUYTOCLOSE
<OPTSELLTYPE>		For options, type of sell: SELLTOCLOSE, SELLTOOPEN
<POSTYPE>		Position type. LONG, SHORT
<RELFITID>		ID of related trade, ***FITID***
<RELTYPE>		Related option transaction type: SPREAD, STRADDLE, NONE, OTHER
<REVERSALFEES>	V2 only	For a reversal transaction, the total amount of fees charged to perform the reversal. ***Amount.***
<REVERSALFITID>	V2 only	For a reversal transaction, the FITID of the transaction that is being reversed. For example, the FITID of the original Buy order. ***FITID.***
<SECURED>		How an option is secured: NAKED, COVERED
<SELLREASON>		Reason the sell of a debt security was generated: CALL (the debt was called), SELL (the debt was sold), MATURITY(the debt reached maturity)
<SELLTYPE>		Type of sell. SELL, SELLSHORT
<SHPERCTRCT>		For options, number of shares per contract, ***N-5***
<SUBACCTFROM>		Sub-account that security or cash is being transferred from: CASH, MARGIN, SHORT, OTHER
<SUBACCTFUND>		Where did the money for the transaction come from or go to? CASH, MARGIN, SHORT, OTHER
<SUBACCTSEC>		Sub-account type for the security: CASH, MARGIN, SHORT, OTHER
<SUBACCTTO>		Sub-account that security or cash is being transferred to: CASH, MARGIN, SHORT, OTHER
<TOTAL>		Transaction total. Buys, sells, etc.:((quan. * (price +/- markup/markdown)) +/- (commission + fees + load + taxes)). Distributions, interest, margin interest, misc. expense, etc.: amount. Return of cap: cost basis; ***amount***
<TAXES>		Taxes on the trade, ***amount***

Tag	Version	Description
<TAXEXEMPT>		Tax-exempt transaction, ***Boolean***
<TFERACTION>		Action for transfers: IN, OUT
<UNITPRICE>		Price per commonly-quoted unit. Does not include markup/markdown, ***unitprice***. Share price for stocks, mutual funds, and others Percentage of par for bonds Per share (not contract) for options
<UNITS>		For security-based actions other than stock splits, ***quantity*** Shares for stocks, mutual funds, and others. Face value for bonds. Contracts for options.
<UNITTYPE>		Type of the units value: SHARES, CURRENCY
<WITHHOLDING>		Tax withholdings, ***amount***

13.9.2.4.3 Investment Buy/Sell Aggregates <INVBUY>/<INVSELL>

These aggregates are referenced within investment transaction aggregates.

Aggregate Name	Version	Elements
INVBUY		**<INVTRAN>** aggregate
		<SECID> aggregate
		<UNITS>
		<UNITPRICE>
		<MARKUP>
		<COMMISSION>
		<TAXES>
		<FEES>
		<LOAD>
		<TOTAL>
	V2 only	<REVERSALFEES>
	V2 only	**<REVERSALFITID>**
		<CURRENCY> aggregate
		<ORIGCURRENCY> aggregate
		<SUBACCTSEC>
		<SUBACCTFUND>
INVSELL		**<INVTRAN>** aggregate
		<SECID> aggregate
		<UNITS>
		<UNITPRICE>
		<MARKDOWN>
		<COMMISSION>
		<TAXES>
		<FEES>
		<LOAD>
		<WITHHOLDING>
		<TAXEXEMPT>
		<TOTAL>
	V2 only	<REVERSALFEES>
	V2 only	<REVERSALFITID>
		<GAIN>
		<CURRENCY> aggregate
		<ORIGCURRENCY> aggregate
		<SUBACCTSEC>
		<SUBACCTFUND>

13.9.2.4.4 Investment Transaction Aggregates

Aggregate Name	Elements	Description
<BUYDEBT>	**<INVBUY>** aggregate	Buy debt security

Aggregate Name	*Elements*	*Description*
	<ACCRDINT>	Accrued interest. This amount is not reflected in the <TOTAL> field of a containing aggregate.
<BUYMF>	**<INVBUY>** aggregate **<BUYTYPE>** <RELFITID>	Buy mutual fund The BUYTOCOVER buy type used to close short sales. RELFITID used to relate transactions associated with mutual fund exchanges.
<BUYOPT>	**<INVBUY>** aggregate **<OPTBUYTYPE>** **<SHPERCTRCT>**	Buy option The BUYTOOPEN buy type is like "ordinary" buying of option and works like stocks.
<BUYOTHER>	**<INVBUY>** aggregate **<BUYTYPE>**	Buy other security type Added for OFX 1.5.
<BUYSTOCK>	**<INVBUY>** aggregate **<BUYTYPE>**	Buy stock The BUYTOCOVER buy type used to close short sales.
<CLOSUREOPT>	**<INVTRAN>** aggregate **<SECID>** aggregate **<OPTACTION>** **<UNITS>** **<SHPERCTRCT>** **<SUBACCTSEC>** <RELFITID> <GAIN>	Close a position for an option. The EXERCISE action is used to close out an option that is exercised. The ASSIGN action is used when an option writer is assigned. The EXPIRE action is used when the option's expired date is reached. When the action is EXERCISE or ASSIGN another transaction must be generated by the server to represent the buy or sell of the underlying security. RELFITID refers to the transaction ID of the underlying buy or sell.
<INCOME>	**<INVTRAN>** aggregate **<SECID>** aggregate **<INCOMETYPE>** **<TOTAL>** **<SUBACCTSEC>** **<SUBACCTFUND>** <TAXEXEMPT> <WITHHOLDING> <CURRENCY> aggregate <ORIGCURRENCY> aggregate	Investment income is realized as cash into the investment account. A negative TOTAL is used to denote adjustments to income.
<INVEXPENSE>	**<INVTRAN>** aggregate <**SECID**> aggregate <**TOTAL**> <**SUBACCTSEC**> <**SUBACCTFUND**> <CURRENCY> aggregate <ORIGCURRENCY> aggregate	Misc. investment expense that is associated with a specific security. If the expense is associated with the account then an INVBANKTRAN - DEBIT should be used.
<JRNLFUND>	**<INVTRAN>** aggregate **<SUBACCTTO>** **<SUBACCTFROM>** **<TOTAL>**	Journaling cash holdings between sub-accounts within the same investment account.
<JRNLSEC>	**<INVTRAN>** aggregate **<SECID>** aggregate **<SUBACCTTO>** **<SUBACCTFROM>** **<UNITS>**	Journaling security holdings between sub-accounts within the same investment account.
<MARGININTEREST>	**<INVTRAN>** aggregate **<TOTAL>** **<SUBACCTFUND>** <CURRENCY> aggregate <ORIGCURRENCY> aggregate	Margin interest expense
<REINVEST>	**<INVTRAN>** aggregate **<SECID>** aggregate **<INCOMETYPE>** **<TOTAL>** **<SUBACCTSEC>** **<UNITS>**	Reinvestment of income REINVEST is a single transaction that contains both income and an investment transaction. If servers can't track this as a single transaction they should return an INCOME transaction

Aggregate Name	Elements	Description
	<UNITPRICE> <COMMISSION> <TAXES> <FEES> <LOAD> <TAXEXEMPT> <CURRENCY> aggregate <ORIGCURRENCY> aggregate	and an INVTRAN. TOTAL and UNITS are signed as for an investment buy. Corrections to a REINVEST are signed as for an investment sell.
<RETOFCAP>	**<INVTRAN>** **<SECID>** **<TOTAL>** **<SUBACCTSEC>** **<SUBACCTFUND>** <CURRENCY> aggregate <ORIGCURRENCY> aggregate	Return of capital
<SELLDEBT>	**<INVSELL>** aggregate **<SELLREASON>** <ACCRDINT>	Sell debt security. Used when debt is sold, called, or reached maturity.
<SELLMF>	**<INVSELL>** aggregate **<SELLTYPE>** <AVGCOSTBASIS> <RELFITID>	Sell mutual fund RELFITID used to relate transactions associated with mutual fund exchanges.
<SELLOPT>	**<INVSELL>** aggregate **<OPTSELLTYPE>** **<SHPERCTRCT>** <RELFITID> <RELTYPE> <SECURED>	Sell option The SELLTOCLOSE action is selling a previously bought option. The SELLTOOPEN action is writing an option
<SELLOTHER>	**<INVSELL>** aggregate <SELLTYPE>	Sell other type of security Added for OFX 1.5. Optional tag.
<SELLSTOCK>	**<INVSELL>** aggregate **<SELLTYPE>**	Sell stock
<SPLIT>	**<INVTRAN>** aggregate **<SECID>** aggregate **<SUBACCTSEC>** **<OLDUNITS>** **<NEWUNITS>** **<NUMERATOR>** **<DENOMINATOR>** <CURRENCY> aggregate <ORIGCURRENCY> aggregate <FRACCASH> <SUBACCTFUND>	Stock or Mutual Fund Split ***Note: the trade date is interpreted as the "day of record" for the split.***
<TRANSFER>	**<INVTRAN>** aggregate **<SECID>** aggregate **<SUBACCTSEC>** **<UNITS>** **<TFERACTION>** **<POSTYPE>** <INVACCTFROM> aggregate <AVGCOSTBASIS> <UNITPRICE> <DTPURCHASE>	Transfer holdings in and out of the investment account.

13.9.2.4.5 Valid Transactions by Security Type

	Debt	Mutual Fund	Option	Other	Stock

	Debt	Mutual Fund	Option	Other	Stock
BUYDEBT	_				
BUYMF		_			
BUYOPT			_		
BUYOTHER				_	
BUYSTOCK					_
CLOSUREOPT			_		
INCOME	_	_		_	_
INVEXPENSE	_	_	_	_	_
JRNLFUND					
JRNLSEC	_	_	_	_	_
MARGININTEREST					
REINVEST	_	_		_	_
RETOFCAP	_	_	_	_	_
SELLDEBT	_				
SELLMF		_			
SELLOPT			_		
SELLOTHER				_	
SELLSTOCK					_
SPLIT		_			_
TRANSFER	_	_	_	_	_

Since JRNLFUND and MARGININTEREST do not refer to securities, there are no checks in any of the security columns for these transactions.

13.9.2.4.6 Notes on Mutual Fund Exchanges

In investment statement download, two transactions are needed to reflect mutual fund exchanges. A SELLMF should be generated for the mutual fund being switched from and a BUYMF should be generated for the mutual fund being switched to. You can use the RELFITID tag to link these two transactions to each other. You should use the MEMO tag of the individual transactions to explain that a mutual fund exchange occurred.

13.9.2.4.7 Notes on Corporate Actions

Since corporate actions can often be very complicated, it is difficult to define a single action aggregate that encompasses all possible scenarios. Instead, you should describe corporate actions using one or more of the provided basic action types. You should use the memo field of the individual transactions to link transactions to an encompassing corporate action.

13.9.2.4.8 Notes on Option Splits

When the underlying security for an option splits, a new security is generated for the option since the strike price changes. In investment statement download, you need two transactions to reflect this activity. There should be a TRANSFER transaction to show that the old option security is removed from the account and another TRANSFER transaction to show that the new option security is moved into the account.

13.9.2.4.9 Notes on Option actions

For options, the overall sequence of actions is as follows:

For an option writer:

Position is opened with Sell to Open.

Position is closed with one of the following:

- Buy to Close
- Expire
- Assigned

For an option buyer:

Position is opened with Buy to Open.

Position is closed with one of the following:

- Sell to Close
- Expire
- Exercise

13.9.2.4.10 Notes on Reversals

The V2 message set introduces an architecture that enables transaction reversals.

To reverse a Buy transaction, an <INVBUY> aggregate is used, with the <REVERSALFITID> containing the <FITID> of the transaction that is being reversed. The number of shares being reversed is contained in <UNITS>; the amount of the fees being refunded is in <FEES>; etc. If a fee is charged to perform the reversal, it appears in <REVERSALFEES>.

13.9.2.5 The same approach is used with reversing a Sell transaction, except that an <INVSELL> aggregate is used instead.Open Orders

13.9.2.5.1 General Open Order Aggregate <OO>

The <OO> aggregate contains fields common to all open orders. Use this aggregate to define the open order aggregates as show in the following section.

Tag	*Version*	*Description*
<OO>		General-open-order aggregate
<FITID>		Unique FI-assigned transaction ID, ***FITID***
<SRVRTID>	V1 only	Unique server-assigned transaction ID, ***SRVRTID***
<SRVRTID2>	V2 only	Unique server-assigned transaction ID, ***SRVRTID2***
<SECID>		Security identified aggregate
</SECID>		
<DTPLACED>		Date-time the order was placed, ***datetime***
<UNITS>		Quantity of the security the open order is for, ***unitprice***
<SUBACCT>		Sub-account type. CASH, MARGIN, SHORT, OTHER
<**DURATION**>		How long the order is good for: DAY, GOODTILCANCEL, IMMEDIATE
<**RESTRICTION**>		Special restriction on the order: ALLORNONE, MINUNITS, NONE
<MINUNITS>		Minimum number of units that must be filled for the order, ***quantity***

Tag	Version	Description
<LIMITPRICE>		Limit price, ***unitprice***
<STOPPRICE>		Stop price, ***unitprice***
<MEMO>	V1 only	Other information about order (at most one), ***memo***
<MEMO2>	V2 only	Other information about order (at most one), ***memo2***
<CURRENCY>		Overriding currency aggregate
</CURRENCY>		
</OO>		

Note: *An open order is assumed to be a market order if no limit price or stop price is specified.*

13.9.2.5.2 Investment Open Order Aggregates

Open Order Aggregates	Elements	Description of Elements
<OOBUYDEBT>	**<OO>** aggregate	
	<AUCTION>	Whether the debt should be purchased at the auction, ***Boolean***
	<DTAUCTION>	Date of the auction, ***date***
<OOBUYMF>	**<OO>** aggregate	
	<BUYTYPE>	Type of purchase: BUY, BUYTOCOVER.
	<UNITTYPE>	What the units represent: SHARES, CURRENCY
<OOBUYOPT>	**<OO>** aggregate	
	<OPTBUYTYPE>	Type of purchase: BUYTOOPEN, BUYTOCLOSE
<OOBUYOTHER>	**<OO>** aggregate	
	<UNITTYPE>	What the units represent: SHARES, CURRENCY
<OOBUYSTOCK>	**<OO>** aggregate	
	<BUYTYPE>	Type of purchase: BUY, BUYTOCOVER
<OOSELLDEBT>	**<OO>** aggregate	
<OOSELLMF>	**<OO>** aggregate	
	<SELLTYPE>	Type of sale: SELL, SELLSHORT
	<UNITTYPE>	What the units represent: SHARES, CURRENCY.
	<SELLALL>	Sell entire holding, ***Boolean***
<OOSELLOPT>	**<OO>** aggregate	
	<OPTSELLTYPE>	Type of sale: SELLTOOPEN, SELLTOCLOSE
<OOSELLOTHER>	**<OO>** aggregate	
	<UNITTYPE>	What the units represent: SHARES, CURRENCY
<OOSELLSTOCK>	**<OO>** aggregate	
	<SELLTYPE>	Type of sale: SELL, SELLSHORT
<SWITCHMF>	**<OO>** aggregate	
	<SECID> aggregate	Security ID of the mutual fund to switch to or purchase
	<UNITTYPE>	What the units represent: SHARES, CURRENCY
	<SWITCHALL>	Switch entire holding, ***Boolean***

13.9.2.6 Investment Positions

Position records represent a user's current positions, regardless of the transactional history. Prices and values should be the most recent available, even if different from a transaction price on the same day.

In position records, securities are identified as being either short or long. Because each FI has different rules regarding which sub-accounts can be used for short compared to long activity, FIs must explicitly indicate the type of position in addition to specifying the sub-account where the position takes place.

For options, position type SHORT is equivalent to WRITING an option, and position type LONG is equivalent to HOLDING an option. For security types where there is only one type (for example, bonds), use LONG.

13.9.2.6.1 General Position Information <INVPOS>

The INVPOS aggregate contains fields relevant to all investment position types. It is included in the position aggregates as shown in the following sections.

Tag	*Version*	*Description*
<INVPOS>		General-position aggregate
<SECID>		Security identifier
</SECID>		
<HELDINACCT>		Sub-account type CASH, MARGIN, SHORT, OTHER
<POSTYPE>		SHORT = Writer for options, Short for all others.
		LONG = Holder for options, Long for all others.
<UNITS>		For stocks, MFs, other, number of shares held. Bonds = face value. Options = number of contracts ***quantity***
<UNITPRICE>		For stocks, MFs, other, price per share. Bonds = percentage of par Option = premium per share of underlying security ***unitprice***
<MKTVAL>		Market value of this position, ***amount***
<DTPRICEASOF>		Date and time of unit price and market value. Can be 0 if unit price and market value are unknown, ***datetime***
<CURRENCY>		Currency information if different from default currency.
</CURRENCY>		
<MEMO>	V1 only	Comment, ***memo***
<MEMO2>	V2 only	Comment, ***memo2***
</INVPOS>		

13.9.2.6.2 Investment Positions

Investment Position Aggregates	*Elements*	*Description of Elements*
<POSDEBT>	**<INVPOS>** aggregate	
<POSMF>	**<INVPOS>** aggregate	
	<UNITSSTREET>	Units in the FI's street name, ***positive quantity***

Investment Position Aggregates	*Elements*	*Description of Elements*
	<UNITSUSER>	Units in the user's name directly, ***positive quantity***
	<REINVDIV>	Reinvest dividends, ***Boolean***
	<REINVCG>	Reinvest capital gains, ***Boolean***
<POSOPT>	**<INVPOS>** aggregate	
	<SECURED>	How the option is secured. NAKED, COVERED.
<POSOTHER>	**<INVPOS>** aggregate	
<POSSTOCK>	**<INVPOS>** aggregate	
	<UNITSSTREET>	Units in the FI's street name, ***positive quantity***
	<UNITSUSER>	Units in the user's name directly, ***positive quantity***
	<REINVDIV>	Reinvest dividends, ***Boolean***

13.9.2.7 Investment Balances <INVBAL>

The <INVBAL> aggregate contains five specified balances. It can also contain a <BALLIST> aggregate that contains one or more <BAL> aggregates. The <BAL> aggregate (see Chapter 3, "Common Aggregates, Elements, and Data Types") allows an FI to send any number of balances to the user, complete with description and Help text. The intent is to capture the same type of balance information present on the first page of many FI brokerage statements. You can also use the <BAL> aggregate to send margin call information.

Tag	*Description*
<INVBAL>	Balances aggregate
<AVAILCASH>	Cash balance across all sub-accounts. Should include sweep funds. ***Amount***
<MARGINBALANCE>	Margin balance. A positive balance indicates a positive cash balance, while a negative balance indicates the customer has borrowed funds. ***Amount***
<SHORTBALANCE>	Market value of all short positions, ***amount***
<BUYPOWER>	Buying power, ***amount***
<BALLIST>	Beginning of Investment balance list (at most one)
<BAL>	Balance aggregates (one or more)
</BAL>	See Chapter 3, "Common Aggregates, Elements, and Data Types"
</BALLIST>	
</INVBAL>	

13.9.2.8 Status Codes

Code	*Meaning*
0	Success
2000	General error (ERROR)
2002	General account error (ERROR)
2003	Account not found (ERROR)
2004	Account closed (ERROR)
2005	Account not authorized (ERROR)
2019	Duplicate request (ERROR)
12250	Investment transaction download not supported (WARN)
12251	Investment position download not supported (WARN)

Code	Meaning
12252	Investment positions for specified date not available (WARN)
12253	Investment open order download not supported (WARN)
12254	Investment balances download not supported (WARN)

13.10 Investment E-Mail

Open Financial Exchange currently defines one investment e-mail message that clients can send to an FI. With this message, the user can prepare a message to the FI regarding one of their accounts. The server acknowledges receipt of the message. The FI prepares the response that the client picks up when it synchronizes with the server. E-mail is subject to synchronization, using <INVMAILSYNCRQ> / <INVMAILSYNCRS>.

Client Sends	Server Responds
Addressed message	
Inv. account information	
	Acknowledgment
.	
.	
.	
Synchronization request	
	Response to customer

13.10.1 Investment E-Mail Request and Response

13.10.1.1 Request <INVMAILRQ>

The client must identify to which investment account the customer query is related.

Tag	Description
<INVMAILRQ>	Investment-e-mail-request aggregate
<INVACCTFROM>	Account-from aggregate
</INVACCTFROM>	
<MAIL>	To, from, message information, see section 9.2.2
</MAIL>	
</**INVMAILRQ**>	

13.10.1.2 Response <INVMAILRS>

Tag	Description
<INVMAILRS>	Investment-e-mail-response aggregate
<INVACCTFROM>	Account-from aggregate
</INVACCTFROM>	
<MAIL>	To, from, message information, see section 9.2.2
</MAIL>	
</**INVMAILRS**>	

13.10.1.3 Status Codes

Code	Meaning
0	Success (INFO)
2000	General error (ERROR)
2002	General account error (ERROR)
2003	Account not found (ERROR)
2004	Account closed (ERROR)
2005	Account not authorized (ERROR)
2019	Duplicate request (ERROR)
16500	HTML not allowed (ERROR)
16501	Unknown mail To: (ERROR)

13.10.2 Investment E-Mail Synchronization

13.10.2.1 Request <INVMAILSYNCRQ>

Tag	Version	Description
<INVMAILSYNCRQ>		Synchronization-request aggregate
Client synchronization option; <TOKEN>, <TOKEN2>, <TOKENONLY>, or <REFRESH>		
<TOKEN>	V1 only	Previous value of <TOKEN> received for this type of synchronization request from server; 0 for first-time requests; ***token***
<TOKEN2>	V2 only	Previous value of <TOKEN2> received for this type of synchronization request from server; 0 for first-time requests; ***token2***
<TOKENONLY>		Request for just the current <TOKEN> without the history, ***Boolean***
<REFRESH>		Request for refresh of current state, ***Boolean***
<REJECTIFMISSING>		If Y, do not process requests if client <TOKEN> is out of date, ***Boolean***
<INCIMAGES>		Y if the client accepts mail with images in the message body. N if the client does not accept mail with images in the message body. ***Boolean***
<USEHTML>		Y if client wants an HTML response, N if client wants plain text, ***Boolean***
<INVACCTFROM>		Investment account of interest; token must be interpreted in terms of this account
</INVACCTFROM>		
<INVMAILTRNRQ>		Investment-mail transactions (0 or more)
</INVMAILTRNRQ>		
</INVMAILSYNCRQ>		

13.10.2.2 Response <INVMAILSYNCRS>

Tag	Version	Description
<INVMAILSYNCRS>		Synchronization-response aggregate
<TOKEN>	V1 only	New synchronization token, ***token***
<TOKEN2>	V2 only	New synchronization token, ***token2***
<LOSTSYNC>		Y if the token in the synchronization request is older than the earliest entry in the server's history table. In this case, some

Tag	Version	Description
		responses have been lost. N if the token in the synchronization request is newer than or matches a token in the server's history table. ***Boolean***
<SYNCERROR>	V2 only	Optional error code, ***N-6***
<INVACCTFROM>		Investment account of interest; token must be interpreted in terms of this account
</INVACCTFROM>		
<INVMAILTRNRS>		Investment-mail transactions (0 or more)
</INVMAILTRNRS>		
</INVMAILSYNCRS>		

13.11 Complete Example

This example is for a user who requests an investment statement download for a single account.

The request file:

```
<OFX>                                       <!--Beginning of request data-->
 <SIGNONMSGSRQV1>
   <SONRQ>                                  <!--Beginning of signon-->
      <DTCLIENT>19960828101000              <!--Datetime, 8/28/96, 10:10:00 am-->
      <USERID>123-45-6789                   <!--User ID (e.g. SSN) -->
      <USERPASS>MYPASSWORD                  <!--Password(SSL encrypts whole) -->
      <LANGUAGE>ENG                         <!--Use English for the response -->
      <APPID>MyApp                          <!--Client application ID-->
      <APPVER>0500                          <!--Version 5.0-->
   </SONRQ>         <!--End of signon-->
 </SIGNONMSGSRQV1>
 <INVSTMTMSGSRQV1>
   <INVSTMTTRNRQ>   <!--First request in file-->
      <TRNUID>1001                          <!--Unique ID for this request-->
      <INVSTMTRQ>   <!--Beginning of statement download-->
         <INVACCTFROM>                      <!--Identify the account: -->
            <BROKERID>121099999             <!--FI ID-->
            <ACCTID>999988                  <!--Account number-->
         </INVACCTFROM>                     <!--End of account ID-->
         <INCTRAN>                          <!--Request transactions-->
            <DTSTART>19960824130105         <!--Send transactions posted after-->
                                            <!--Aug 24, 1996 1:01:05pm-->

            <INCLUDE>Y                      <!--Include transactions -->
         </INCTRAN>
         <INCOO>Y   <!--Include open orders in response-->
         <INCPOS>   <!--Request positions -->
           <INCLUDE>Y                       <!--Include current positions -->
         </INCPOS>
         <INCBAL>Y                          <!--Include balances in request-->
      </INVSTMTRQ>
   </INVSTMTTRNRQ>                          <!--End of first request-->
 </INVSTMTMSGSRQV1>
</OFX>                                      <!--End of Open Financial Exchange
                                            request data-->
```

A typical server response:

This user has one investment transaction, one bank transaction, one open order, two position entries, and one balance entry. The user deposits some money and buys shares in Acme. The user has an open limit order to buy 100 shares of Hackson Unlimited at $50/share. The holdings show the user already had 100 shares of Acme and now has 200 shares. The user also has one option contract to sell Lucky Airlines shares, bought before this download.

```
<OFX>                                   <!--Beginning of request data-->
 <SIGNONMSGSRSV1>
   <SONRS>                              <!--Beginning of signon-->
     <STATUS>
         <CODE>0                        <!--0 = OK-->
         <SEVERITY>INFO
      </STATUS>
      <DTSERVER>19960828101003          <!--Timestamp, 8/28/96, 10:10:03 am-->
      <LANGUAGE>ENG                     <!--English is used in this response-->
      <DTPROFUP>19960101120000          <!--Profile updated, 1/1/96, 12 am-->
      <DTACCTUP>19960101120000          <!--Account info updated,1/1/96,12am-->
   </SONRS>         <!--End of signon-->
</SIGNONMSGSRSV1>
<INVSTMTMSGSRSV1>
   <INVSTMTTRNRS>   <!--First request in file-->
      <TRNUID>1001                      <!--Client ID for this request-->
      <STATUS>
         <CODE>0                        <!--0 = accepted, good data follows-->
         <SEVERITY>INFO
      </STATUS>
      <INVSTMTRS>   <!--Beginning of statement download-->
         <DTASOF>19960827010000         <!--Statement as of Aug 27, 1996 1am-->
         <CURDEF>USD <!--Default currency is US Dollar-->
         <INVACCTFROM>                  <!--Beginning of account information-->
            <BROKERID>121099999         <!--FI ID-->
            <ACCTID>999988              <!--Account number-->
         </INVACCTFROM>                 <!--End of account information-->
         <INVTRANLIST>                  <!--Beginning of transactions-->
            <DTSTART>19960824130105     <!--Send transactions posted after-->
                                        <!--Aug 24, 1996 1:01:05pm-->
            <DTEND>19960828101000       <!--End timestamp (now) -->
            <BUYSTOCK>                  <!--Buy stock transaction-->
               <INVBUY>
                  <INVTRAN>
                     <FITID>23321       <!--FI transaction ID-->
                     <DTTRADE>19960825  <!--Trade date Aug 25, 1996-->
                     <DTSETTLE>19960828 <!--Settlement date Aug 28, 1996-->
                  </INVTRAN>
                  <SECID>               <!--Security ID-->
                     <UNIQUEID>123456789 <!--CUSIP for ACME -->
                     <UNIQUEIDTYPE>CUSIP
                  </SECID>
                  <UNITS>100            <!--100 shares-->
                  <UNITPRICE>50.00      <!--$50/share-->
                  <COMMISSION>25.00     <!--$25 commission -->
                  <TOTAL>5025.00        <!--Total amount $5025.00-->
                  <SUBACCTSEC>CASH      <!--Holding resides in cash account-->
                  <SUBACCTFUND>CASH     <!--Bought in cash account-->
               </INVBUY>
               <BUYTYPE>BUY             <!--Normal buy-->
            </BUYSTOCK>                 <!--End of buy stock transaction-->
            <INVBANKTRAN>               <!--Investment acct bank transaction-->
               <STMTTRN>                <!--Beginning of a bank transaction-->
                  <TRNTYPE>CREDIT       <!--Generic credit-->
                  <DTPOSTED>19960825    <!--Aug 25, 1996-->
                  <DTUSER>19960825      <!--Aug 25, 1996-->
                  <TRNAMT>1000.00       <!--$1,000.00-->
                  <FITID>12345          <!--FI transaction ID 12345-->
                  <NAME>Customer deposit <!--Description of transaction-->
```

```
            <MEMO>Your check #1034    <!--Optional memo from FI-->
         </STMTTRN>                   <!--End of bank transaction-->
         <SUBACCTFUND>CASH            <!--Credited to the cash account -->
      </INVBANKTRAN>
   </INVTRANLIST>                     <!--End of transactions-->
   <INVPOSLIST><!--Beginning of positions list-->
      <POSSTOCK>                      <!--Beginning of position -->
         <INVPOS>
            <SECID>                   <!--Security ID-->
               <UNIQUEID>123456789    <!--CUSIP for Acme Development, Inc.-->
               <UNIQUEIDTYPE>CUSIP
            </SECID>
            <HELDINACCT>CASH          <!--Cash account-->
            <POSTYPE>LONG             <!--Long position-->
            <UNITS>200                <!--200 shares-->
            <UNITPRICE>49.50          <!--Latest price-->
            <MKTVAL>9900.00           <!--Current market value $9900.00-->
            <DTPRICEASOF>19960827010000 <!--Prices as of Aug27,1996 1am-->
            <MEMO>Next dividend payable Sept 1
         </INVPOS>
      </POSSTOCK>                     <!--End of position-->
      <POSOPT> <!--Beginning of position-->
         <INVPOS>
            <SECID>                   <!--Security ID-->
               <UNIQUEID>000342222    <!--CUSIP for the option -->
               <UNIQUEIDTYPE>CUSIP
            </SECID>
            <HELDINACCT>CASH          <!--Cash account-->
            <POSTYPE>LONG             <!--Long position-->
            <UNITS>1                  <!--100 shares-->
            <UNITPRICE>5              <!--Latest price-->
            <MKTVAL>500               <!--Current market value $500.00-->
            <DTPRICEASOF>19960827010000 <!--Prices as of Aug27,1996 1am-->
            <MEMO> Option is in the money
         </INVPOS>
      </POSOPT>                       <!--End of option position -->
   </INVPOSLIST>                      <!--End of position -->
   <INVBAL>
      <AVAILCASH>200.00               <!--$200.00 cash balance-->
      <MARGINBALANCE>-50.00           <!--$50.00 owed on margin balance-->
      <SHORTBALANCE>0                 <!--$0 short balance-->

      <BALLIST>                       <!--Beginning of FI-defined balances-->
         <BAL>                        <!--Beginning of a balance-->
            <NAME>Margin Interest Rate <!--Name of balance entry-->
            <DESC>Current interest rate on margin balances
                                      <!--Help text for this balance-->
            <BALTYPE>PERCENT          <!--Format as percent-->
            <VALUE>7.85               <!--Will be formatted 7.85%-->
            <DTASOF>19960827010000    <!--Rate as of Aug 27, 1996 1am-->
         </BAL>                       <!--End of balance entry-->
      </BALLIST>                      <!--End of balances-->
   </INVBAL>
   <INVOOLIST>
     <OOBUYSTOCK>
         <OO>
            <FITID>23321              <!--FI transaction ID-->
            <SECID>                   <!--Security ID-->
               <UNIQUEID>666678578    <!--CUSIP for Hackson Unlimited-->
               <UNIQUEIDTYPE>CUSIP
            </SECID>
            <DTPLACED>19960624031505<!--Order placed 6/24/96 3:15:05pm-->
            <UNITS>100                <!--100 shares-->
            <SUBACCT>CASH             <!--Purchase with cash-->
```

```
                <DURATION>GOODTILCANCEL   <!--GOODTILCANCEL-->
                <RESTRICTION>NONE         <!--No special restrictions-->
             <LIMITPRICE>50.00            <!--Limit price $50/share-->
             </OO>
             <BUYTYPE>BUY                 <!--Normal buy-->
          </OOBUYSTOCK>
       </INVOOLIST>
    </INVSTMTRS>
  </INVSTMTTRNRS>                         <!--End of first response-->
</INVSTMTMSGSRSV1>
<SECLISTMSGSRSV1>
  <SECLIST>                               <!--Beginning of securities list-->
     <STOCKINFO>   <!--Beginning of 1st security ID-->
        <SECINFO>
           <SECID>                        <!--Security ID-->
              <UNIQUEID>123456789         <!--CUSIP for the stock -->
              <UNIQUEIDTYPE>CUSIP
           </SECID>
           <SECNAME>Acme Development, Inc.
           <TICKER>ACME                   <!--Ticker symbol-->
           <FIID>1024                     <!--FI internal security identifier-->
        </SECINFO>
        <YIELD>10                         <!--$10/share/year-->
        <ASSETCLASS>SMALLSTOCK            <!--Small Capital Stock asset class-->
     </STOCKINFO>                         <!--End of security ID-->
     <STOCKINFO>
        <SECINFO>
           <SECID>                        <!--Security ID-->
              <UNIQUEID>666678578         <!--CUSIP for the stock -->
              <UNIQUEIDTYPE>CUSIP
           </SECID>
           <SECNAME> Hackson Unlimited, Inc.
           <TICKER> HACK                  <!--Ticker symbol-->
           <FIID>1027                     <!--FI internal security identifier-->
        </SECINFO>
        <YIELD>17                         <!--$10/share/year-->
        <ASSETCLASS>SMALLSTOCK            <!--Small Capital Stock asset class-->
     </STOCKINFO>
     <OPTINFO>                            <!--End of security ID-->
        <SECINFO>
           <SECID>                        <!--Security ID-->
              <UNIQUEID>000342222         <!--CUSIP for the option -->
              <UNIQUEIDTYPE>CUSIP
           </SECID>
           <SECNAME>Lucky Airlines Jan 97 Put
           <TICKER>LUAXX                  <!--Ticker symbol-->
           <FIID>0013                     <!--FI internal security identifier-->
        </SECINFO>
        <OPTTYPE>PUT
        <STRIKEPRICE>35.00                <!--Strike price $35/share-->
        <DTEXPIRE>19970121                <!--Option expires Jan 21, 1997-->
        <SHPERCTRCT>100                   <!--100 shares per contract-->
        <SECID>                           <!--Security ID-->
           <UNIQUEID>000342200            <!--CUSIP for the underlying stock -->
           <UNIQUEIDTYPE>CUSIP
        </SECID>
        <ASSETCLASS>LARGESTOCK            <!--Large Capital Stock asset class-->
     </OPTINFO>                           <!--End of option information-->
  </SECLIST>                              <!--End of securities list-->
</SECLISTMSGSRSV1>
</OFX>                                    <!--End of Open Financial Exchange
                                          request data-->
```

14. Bill Presentment

14.1 Overview

Bill Presentment (PRES) is the electronic delivery of a bill from a biller to a customer.

Although some billers may provide Bill Presentment service themselves, many will choose to work with a bill publisher that provides Bill Presentment service on behalf of many billers. For this reason, Bill Presentment focuses on connecting customers to bill publishers.

14.1.1 Bill Presentment Model

This section summarizes the process of receiving bills electronically, starting with the steps required to find a bill publisher and set up Bill Presentment service.

To receive bills electronically, the client:

Finds one or more billers by searching a biller directory server.

Determines which bill publishers provide Bill Presentment service for the billers.

Enrolls with a bill publisher for Bill Presentment service

Signs on with the bill publisher and activates Bill Presentment service for one or more accounts with one or more billers.

Requests electronic bills from the bill publisher.

(Optionally) Pays bills using the Open Financial Exchange Bill Payment service.

14.1.2 Servers and Message Sets

During the billing process, the client typically communicates with two Open Financial Exchange servers:

- *Biller directory server:* An independent server that stores information about billers and bill publishers. Clients can query this server to find the bill publishers that serve the billers in which the customer is interested.
- *Bill publisher server:* The server that delivers bills to customers. A single bill publisher can provide Bill Presentment service for many billers. In some cases, a biller might act as its own bill publisher.

Although it is possible for a single server to perform both of the functions listed above, it is more likely that independent directory servers will provide clients with a single source for finding billers. To allow these functions to be routed separately by clients, Bill Presentment defines separate message sets for directory query and bill delivery.

- Biller Directory message set <PRESDIRMSGSETV1>
- Bill Delivery message set <PRESDLVMSGSETV1>

For additional information about the message sets defined for Bill Presentment, see section 14.7.

14.2 Biller Directory

To find billers, the client sends a <FINDBILLERRQ> request to the biller directory server. The biller directory server returns a <FINDBILLERRS> response.

<FINDBILLERRQ> and <FINDBILLERRS> are part of the Biller Directory message set <PRESDIRMSGSETV1>. The message set tags are <PRESDIRMSGSRQV1> and <PRESDIRMSGSRSV1>.

14.2.1 Client Signon to the Biller Directory Server

Because the client does not enroll with the biller directory server, the client sends a standard signon request with dummy values for the user ID and password. Only the basic client identification will be valid. For more information about signon requests, refer to Chapter 2, "Structure."

14.2.2 Search Arguments

If the client omits all elements in the <FINDBILLERRQ>, the client is requesting a complete directory of billers. Otherwise, the client wants to filter results based on the included elements, with strings matched case-insensitive using regular expressions (for example, '*' is interpreted as a wildcard).

For each biller that matches the elements in the request, the biller directory server returns the complete name and address of the biller, plus the biller ID and bill publisher name.

14.2.3 Identification of Bill Publishers

Bill Publishers must be uniquely and consistently identified by name. Clients need some way to relate the bill publisher name given by a directory server to their own databases of known and approved bill publishers. Since the number of bill publishers is relatively small, and the number of directory servers that must be coordinated even smaller, the official corporate name of a bill publisher will serve as an ID for that publisher.

14.2.4 Find Biller Request <FINDBILLERRQ>

The <FINDBILLERRQ> request must appear within a <FINDBILLERTRNRQ> transaction wrapper.

Clients that want to receive an updated list of billers and bill publishers can use <DTUPDATE> to avoid receiving a response if nothing has changed. <DTUPDATE> is returned by servers in responses to indicate the date and time of the newest or most recently changed entry, whether or not it was included in the response. Clients performing narrow searches cannot use <DTUPDATE> unless they save the value for each query, and send the corresponding value in future requests.

The name and address fields refer to the biller, except for <CONSUPOSTALCODE> which refers to the customer's address.

Tag	*Description*
<FINDBILLERRQ>	
<DTUPDATE>	Date and time of last change to any biller entry as reported by the server on previous query, ***datetime.***
<BILLERID>	ID of this biller at this bill publisher, ***A-32***
<NAME>	Biller's name, regular expression supported, ***A-32***
<ADDR1**>**	Biller's address line 1, regular expression supported, ***A-32***
<ADDR2**>**	Biller's address line 2, regular expression supported, ***A-32***
<ADDR3**>**	Biller's address line 3, regular expression supported, ***A-32***

Tag	Description
<CITY>	Biller's city, regular expression supported, ***A-32***
<STATE>	Biller's state, regular expression supported, ***A-5***
<POSTALCODE>	Biller's postal code, regular expression supported, ***A-11***
<COUNTRY>	ISO/DIS-3166 3-letter country code standard, ***A-3***
<SIC>	Standard Industry Code, ***N-6***
<CONSUPOSTALCODE>	Postal code of customer, to allow server to filter out billers that do not do business in the customer's area, ***A-11***
<INCIMAGES>	Y if the client wants images (logos) returned, ***Boolean***
</FINDBILLERRQ>	

14.2.5 Find Biller Response <FINDBILLERRS>

<FINDBILLERRS> must appear within a <FINDBILLERTRNRS> transaction wrapper.

The response is a list of <BILLERINFO> aggregates.

Tag	Description
<FINDBILLERRS>	
<**DTUPDATE**>	Date and time of last addition or modification to the entries in the directory, whether part of this response or not, ***datetime***
<BILLERINFO>	Zero or more <BILLERINFO> aggregates
</BILLERINFO**>**	
</FINDBILLERRS>	

14.2.5.1 Biller Information <BILLERINFO>

<BILLERINFO> includes information about a single biller.

Besides basic name and address information, <BILLERINFO> includes the <BILLPUB> and <BILLERID> elements. These elements will be used with the customer's account number to identify the customer's account with the biller. For more information about the account-identification aggregates, refer to <PRESACCTFROM> and <PRESACCTTO> in section 14.3.2.2.

<BILLERINFO> can optionally include elements that specify the format of valid account numbers. <ACCTFORMAT> and <ACCTEDITMASK> provide information to the client. <HELPMESSAGE> provides a text message that the client can display to the customer.

To avoid the complications caused by invalid account numbers, <BILLERINFO> can also include a <VALIDATE>*URL* element that the client application can use to validate the customer's account number. See section 14.2.7 for more detail on this.

Tag	Version	Description
<BILLERINFO>		
<**BILLPUB**>		Official standard name of the bill publisher, ***A-32***
<**BILLERID**>		ID of this biller at this bill publisher, ***A-32***
<**NAME**>		Name of the biller, ***A-32***
<ADDR1>		Biller's address line 1, ***A-32***
<ADDR2>		Biller's address line 2, ***A-32***
<ADDR3>		Biller's address line 3, ***A-32***
<CITY>		Biller's city, ***A-32***

Tag	Version	Description
<STATE>		Biller's state, ***A-5***
<POSTALCODE>		Biller's postal code, ***A-11***
<COUNTRY>		Biller's country; 3-letter country code from ISO/DIS-3166, ***A-3***
<SIC>		Standard Industrial Classification Code, ***N-6***
<**PHONE**>		Biller's phone number for customer information (if a special number exists for electronic bilding information, use that number), ***A-32***
<PAYMENTINSTRUMENTS>		Types of payment that the biller can accept electronically, see section 14.3.5.1
</PAYMENTINSTRUMENTS>		
<ACCTFORMAT**>**		Regular expression describing the account number format, ***A-255***
<ACCTEDITMASK**>**		String describing the edit mask for the account number. The client can use the edit mask to assist the user in entering the account number, ***A-255***
<HELPMESSAGE>		Human-readable message that the client can display to assist the customer in entering his or her account number, ***A-255***
<RESTRICT**>**		Human-readable description of any restrictions on who may sign up with this biller. ***A-255***
<LOGO**>**		URL of the biller's logo. If the client requested images, the logo should be included via multi-part MIME in this response. ***URL2***
<VALIDATE>		URL for validation. The client application may use this to validate the customer's account number, see section 14.2.7. ***URL2***
<BILLERINFOURL>		URL of human-readable description of additional information the biller would like the customer to have with regard to signing up. ***URL2***
</BILLERINFO>		

14.2.6 Status Codes <FINDBILLERTRNRS>

Code	Meaning
0	Success (INFO)
2000	General error (ERROR)

14.2.7 Account Number Validation

Servers should implement a lightweight CGI (or equivalent) to validate account numbers. The URL provided in the <VALIDATE> can be accessed with an HTTP GET with three arguments: BILLERID, ACCOUNTNUMBER and CUSTOMERPOSTALCODE. The URL should respond with a text file that includes the following values:

1) Status: (Mandatory)

   Error: An error condition (wrong number of parameters, Database error, etc.). Clarifying text may accompany the error status.

   Passes: The account number is in an acceptable form for this biller (this is not a guarantee that the account will be accepted for the service).

   Failed: The account number does not correspond to an acceptable account number for this biller. Clarifying text may accompany the failed status.

2) Account: (Optional) The preferred format or version of the account number presented in the request.

Heading: (Optional) Additional text to help explain problems to end-users.

Example:

<VALIDATE> = http://testit.com/validate.cgi

Client application uses HTTP GET with
" http://testit.com/validate.cgi?billerid=5454&accountnumber=123-456-7890&customerpostalcode=12345"

The server would respond with one of these:

1) Error

```
Content-type: text/plain

<STATUS>error

<HEADING>The server is unable to process your request at this
time. Please resubmit.
```

2) Failure

```
Content-type: text/plain

<STATUS>failed

<HEADING>123-456-7890 does not appear to be a valid account
number
```

3) Passed

```
Content-type: text/plain
<STATUS>passed
<ACCOUNT>1234567890
```

14.3 Customer Signup

Once the customer has located a biller and its associated bill publisher, the customer must enroll with the bill publisher for Bill Presentment service and activate accounts for one or more billers at that bill publisher.

Bill Presentment uses the standard Open Financial Exchange Signup message set. This section discusses only those portions of signup that differ for Bill Presentment. For more information about the Signup message set, refer to Chapter 8, "Activation & Account Information."

14.3.1 Enrollment

To enroll with a bill publisher, the client uses the standard Open Financial Exchange enrollment aggregate <ENROLLRQ>. The bill publisher server returns an <ENROLLRS> that provides status about the enrollment and optionally returns a user ID and password to be used during subsequent signons.

To allow a client proxy system to enroll multiple clients, Bill Presentment allows multiple <ENROLLRQ> requests in the same Open Financial Exchange file, each wrapped in a <ENROLLTRNRQ>. A given bill publisher may not need some required fields in the <ENROLLRQ>; in such a case, the client proxy system can include dummy data.

14.3.2 Account Inquiry

To receive account information from a bill publisher, the client can use the standard Open Financial Exchange <ACCTINFORQ> aggregate, contained in the <ACCTINFOTRNRQ> wrapper.

The <ACCTINFORS> response returns a <PRESACCTINFO> aggregate for each of the customer's accounts with the billers at that bill publisher. Typically, the response will list only those accounts that have been activated for Bill Presentment service, not all available accounts.

Unlike a financial institution, bill publishers generally won't have information about all the accounts of its supported billers. Billers that also serve as their own bill publishers may be able return available accounts as well as activated accounts. The bill publisher can use the <AVAILACCTS> element in the profile for the Signup message set to indicate whether the server can return available account information.

If the server cannot return information about all available accounts, the client must ask customers for account information prior to requesting service activation for one or more accounts.

14.3.2.1 Bill Presentment Account Information <PRESACCTINFO>

The <PRESACCTINFO> aggregate appears within the <ACCTINFORS> aggregate.

Tag	*Description*
<PRESACCTINFO>	Opening tag for bill presentment account information
<**PRESACCTFROM**>	Bill presentment account identification, see section 14.3.2.2
</PRESACCTFROM>	
<SVCSTATUS>	Status of the Bill Presentment service for this account– AVAIL, PEND, ACTIVE, or REJECTED
<REASON>	Relevant only if REJECTED, ***A-255***
</PRESACCTINFO>	Closing tag for bill presentment account information

14.3.2.2 Account Identification <PRESACCTFROM> <PRESACCTTO>

The <PRESACCTFROM> aggregate uniquely identifies a customer's account with a biller by the combination of bill publisher, biller ID, and account number. Biller IDs must be unique within a bill publisher.

Tag	*Description*
<PRESACCTFROM>	
<**BILLPUB**>	Official standard name of bill publisher, ***A-32***
<**BILLERID**>	ID of this biller at this bill publisher, ***A-32***
<**ACCTID**>	Account number, ***A-32***
<PRESNAMEADDRESS>	Customer's name/address with the biller , see section 14.3.2.2.1
</PRESNAMEADDRESS>	
<USERID>	Customer's user ID, ***A-32***
</PRESACCTFROM>	

<PRESACCTTO> follows the same structure as <PRESACCTFROM>.

14.3.2.2.1 Customer Information with the biller <PRESNAMEADDRESS>

Tag	*Description*
<PRESNAMEADDRESS>	Customer name and address information
<NAMEACCTHELD>	Customer's name as it appears on the account, ***A-96***
<ADDR1>	Customer's address line 1, ***A-32***
<ADDR2>	Customer's address line 2, ***A-32***
<ADDR3>	Customer's address line 3, ***A-32***
<CITY>	Customer's city, ***A-32***
<STATE>	Customer's state, ***A-5***
<POSTALCODE>	Customer's postal code, ***A-11***
<COUNTRY>	Customer's country; 3-letter country code from ISO/DIS-3166, ***A-3***
<DAYPHONE>	Customer's telephone number, ***A-32***
<EVEPHONE>	Customer's telephone number, ***A-32***
</PRESNAMEADDRESS>	

14.3.3 Service Activation

Bill Presentment uses the standard service activation messages defined in Chapter 8, "Activation & Account Information."

The account service request aggregate <ACCTRQ> accepts action-specific aggregates for service additions, changes, and deletions. To add Bill Presentment service to an account, the client sends an <ACCTRQ> with an <SVCADD> for the service <SVC>PRESSVC.

14.3.3.1 Service Addition <SVCADD>

Bill Presentment defines a variant of the service addition aggregate <SVCADD>. This variant contains the <PRESACCTTO> aggregate and <PREAUTHTOKEN> element.

The <PRESACCTTO> aggregate identifies the customer's name and address <PRESNAMEADDRESS>, as it is registered at the biller. Clients are REQUIRED to include the <PRESNAMEADDRESS> aggregate in the <PRESACCTTO> aggregate when they send an <ACCTRQ>.

The <PRESACCTTO> aggregate also identifies the requester's user ID <USERID>. Clients can optionally include a <USERID> that is different from the one used in the <SONRQ>. This <USERID> supports account activation by a third party on behalf of a user. It is up to the server whether to honor such a request, based on access rights granted to the <SONRQ> user. More than one <SVCADD>, on behalf of multiple <USERID>s, can be present in a single <OFX> file.

In some cases, the biller might provide the customer with information out-of-band to expedite service activation. For instance, the biller might provide a confirmation number on the customer's printed statement. The client can provide this additional information through the <PREAUTHTOKEN> element.

Tag	*Description*
<SVCADD>	Opening tag for the service addition
<**PRESACCTTO>**	Account for which Bill Presentment service should be added. See section 14.3.2.2
</PRESACCTTO>	
<PREAUTHTOKEN>	A token provided out-of-band by biller to speed up activation of the account, ***A-32***
</SVCADD>	

14.3.4 Service Status Update for Groups of Customers

The service activation requested with <SVCADD> will often not happen immediately. In this case, a request for account information <ACCTINFORQ> will return an <ACCTINFORS> with a <SVCSTATUS> value of PEND. To find out whether the account has been activated, the client can either send <ACCTINFORQ> once per session until it returns <SVCSTATUS>ACTIVE, or it can include an <ACCTSYNRQ> in each session to catch an unsolicited <ACCTRS> response to the <SVCADD> message.

This section describes a method of checking for status changes on behalf of a group of customers. This method is designed to be used by customer service representatives and client proxy systems.

14.3.4.1 Account Information Request <ACCTINFORQ>

The client uses <ACCTINFORQ> to request information about accounts whose status has changed since the last time the request was made. This request is typically used to retrieve a list of accounts whose status has changed from <SVCSTATUS>PEND to <SVCSTATUS>ACTIVE.

The <ACCTINFORQ> request must appear within a <PRESGRPACCTINFOTRNRQ> transaction wrapper. The transaction wrapper indicates whether the request is for a single user or a group of users.

Tag	*Description*
<ACCTINFORQ>	Opening tag for billing account information request

Tag	Description
<DTACCTUP>	Last <DTACCTUP> received in a response, ***datetime***
</ACCTINFORQ>	Closing tag for billing account information request

14.3.4.2 Account Information Response <ACCTINFORS>

The <ACCTINFORS> aggregate contains zero or more <ACCTINFO> aggregates, which provide the updated account information.

The <ACCTINFORS> response must appear within a <PRESGRPACCTINFOTRNRS> transaction wrapper

Tag	Description
<ACCTINFORS>	Opening tag for billing account information response
<DTACCTUP>	Date and time of last update to account information on the server, ***datetime***
<ACCTINFO>	Zero or more account information aggregates, see section 8.5.3. Each <ACCTINFO> aggregate contains at most one <PRESACCTINFO> aggregate, consistent with section 8.5.3.
</ACCTINFO>	
</ACCTINFORS>	Closing tag for billing account information response

14.3.4.3 Group Account Information Transaction Request <PRESGRPACCTINFOTRNRQ>

As a special transaction wrapper for <ACCTINFORQ>, <PRESGRPACCTINFOTRNRQ> specifies whether the client is requesting account information for a single user or a group of users.

If the client specifies <USERID>, the client is requesting updated account information for a single user. If the user's account information has changed since the last time the client requested it, the server will return the account information in <ACCTINFORS>

If the client specifies <GROUPID>, the client is requesting updated account information for a group of users. The server returns an <ACCTINFORS> response with a <PRESACCTINFO> aggregate for each account whose status has changed.

The client should specify either <USERID> or <GROUPID>; if both are absent, the server uses the <USERID> from the signon request <SONRQ>.

Tag	Description
<PRESGRPACCTINFOTRNRQ>	Opening tag for the transaction request
<**TRNUID**>	Client-assigned globally unique ID for this transaction, ***trnuid***
<CLTCOOKIE>	Data to be echoed in the transaction response, ***A-32***
<TAN>	Transaction authorization number, ***A-80***
<USERID>	Requests account information for the specified user, ***A-32***
- or -	
<GROUPID>	Requests account information for users in the group, ***A-32***
<ACCTINFORQ>	
</ACCTINFORQ>	
</PRESGRPACCTINFOTRNRQ>	Closing tag for the transaction request

14.3.4.4 Group Account Information Transaction Response <PRESGRPACCTINFOTRNRS>

As a special transaction wrapper for <ACCTINFORS>, <PRESGRPACCTINFOTRNRS> contains an <ACCTINFORS> aggregate with zero or more <PRESACCTINFO> aggregates. The <ACCTINFORS> aggregate returns one <PRESACCTINFO> aggregate for each account for which there was a change of status since the <DTACCTUP> date specified.

The server includes information for only those user IDs for which the requester has access rights.

Tag	*Description*
<PRESGRPACCTINFOTRNRS>	Opening tag for the transaction response
<TRNUID>	Client-assigned globally unique ID for this transaction, ***trnuid***
<STATUS>	
</STATUS>	
<CLTCOOKIE>	Data to be echoed in the transaction response, ***A-32***
<ACCTINFORS>	Account information response aggregate
</ACCTINFORS>	
</PRESGRPACCTINFOTRNRS>	Closing tag for the transaction response

14.3.4.5 Status Codes <PRESGRPACCTINFOTRNRS>

Code	*Meaning*
0	Success (INFO)
2000	General error (ERROR)
2002	General account error (ERROR)
2006	Account not found (ERROR)
2008	Account not authorized (ERROR)

14.3.5 Biller Payment Restrictions

In the <PAYMENTINSTRUMENTS> aggregate of <BILLERINFO> aggregate (see section 14.2.5.1), the biller specifies the type of payment instruments it can accept for electronic payment. Since electronic payment does not have to occur through the Open Financial Exchange Bill Payment message set, the payment instruments can include some that Open Financial Exchange doesn't support—for example, CyberCash.

In some cases, a biller may have arranged for a certain party to act as its payment concentrator. This means that the biller expects to receive good funds and remit advice in a pre-arranged format from these concentrators. Such a biller will want to have customers direct their payments to the payment concentrator.

The <PAYMENTINSTRUMENTS> aggregate supports the specification of a payment concentrator from which the biller wants to receive funds. However, Open Financial Exchange currently does not provide a way for clients to get information about payment concentrators. Knowledge of payment concentrators will have to be "hardwired" into client applications. For example, the client application may know that a certain payment concentrator, BigConcentrator, is capable of receiving DigiCash funds. A biller who has BigConcentrator as its payment concentrator can then accept DigiCash funds from the customer without having to support the DigiCash protocol and infrastructure directly. The application would direct the DigiCash funds to the concentrator, who would in turn transfer funds and remit advice to the biller using the agreed-upon method.

14.3.5.1 Payment Instruments <PAYMENTINSTRUMENTS>

In the <PAYMENTINSTRUMENTS> aggregate, billers list which payment instruments they accept.

Tag	*Description*

Tag	Description
<**PAYMENTINSTRUMENTS**>	Opening tag for payment instruments
<PAYMENTINSTRUMENT>	One or more payment instrument aggregates, see section 14.3.5.2
</PAYMENTINSTRUMENT>	
</PAYMENTINSTRUMENTS>	Closing tag for payment instruments

14.3.5.2 Payment Type and Brand <PAYMENTINSTRUMENT>

Each payment instrument is described by <PMTINSTRUMENTTYPE> and <BRAND>. If the server does not specify <BRAND>, the client assumes that all brands of the given <PMTINSTRUMENTTYPE> are acceptable.

Tag	Description
<**PAYMENTINSTRUMENT**>	Opening tag for payment instrument aggregate
<PMTINSTRUMENTTYPE>	Payment type, see section 14.3.5.3
<BRAND>	Accepted brand for given payment type, ***A-32***
</PAYMENTINSTRUMENT>	Closing tag for payment instrument aggregate

14.3.5.3 Payment Instrument Types <PMTINSTRUMENTTYPE>

Type	Description
CONCENTRATOR	Party with which the biller has a business relationship and who will send the biller good funds and remit advice
CHECKINGACCOUNT	Draft on a demand deposit account (US)
CREDITCARD	Payment by Auth/Settle using Credit Card networks
ECOIN	Protocol for payment with electronic cash

If the <BILLERINFO> does not list <PAYMENTINSTRUMENTS>, the following single <PAYMENTINSTRUMENT> is implied:

```
<PAYMENTINSTRUMENT>
   <PMTINSTRUMENTTYPE>CHECKINGACCOUNT
</PAYMENTINSTRUMENT>
```

14.4 Bill Delivery

The Bill Delivery message set contains messages to obtain bills. The message set <PRESDLVMSGSETV1> contains the following tags: <PRESDLVMSGSRQV1> and <PRESDLVMSGSRSV1>.

14.4.1 Bill Delivery Process

Typically, the client periodically requests a list of bills from the bill publisher. The bill publisher responds with a list of bills, each of which contains summary data such as the due date and amount due. For each bill, the bill publisher might also return a URL to a Web site that contains an HTML-rendered version of the bill. Depending on the client's request, the server might also return structured bill detail for a given bill.

The aggregate for a bill list request is <PRESLISTRQ>. This request must be wrapped inside <PRESLISTTRNRQ>. There is no synchronization wrapper for bill list requests, since clients that require a list of bills can send another <PRESLISTRQ>.

The transaction wrapper <PRESLISTTRNRQ> contains optional elements that allow bills for one or more customers to be accessed by customer service representatives or client proxy systems. It is up to the server to decide who can access bills other than their own; it is recommended that all such access be logged in an audit trail.

14.4.2 Bill List Retrieval

<PRESLISTRQ> retrieves bills from the bill publisher. The bill publisher returns a <PRESLISTRS> response that contains a list of one or more bills.

14.4.2.1 Bill List Request <PRESLISTRQ>

The client requests bills from a bill publisher by date range. To specify the date range, clients use <DTSTART> and <DTEND>, as described in section 3.2.7. The request doesn't specify an individual biller; it covers all the billers whose bills are published by the specified bill publisher.

The bill publisher returns information sufficient to identify the biller and provide the amount due, due date, and remittance information so that a payment can be made to the biller. The bill publisher does not provide a viewable form of the bill, but returns a URL to an HTML rendering of the bill. Billing detail, such as individual purchases or transactions, can be included in the original response or obtained from a subsequent <PRESDETAILRQ>.

The <PRESLISTRQ> must be wrapped in the <PRESLISTTRNRQ> transaction wrapper.

Tag	*Description*
<**PRESLISTRQ**>	Opening tag for bill list request
<BILLPUB>	Official standard name of bill publisher, ***A-32***
<DTSTART>	If present, indicates earliest date for which to include bills, ***datetime***
<DTEND>	If present, indicates latest date for which to include bills, ***datetime***
<BILLID>	If present, restrict response to given statement identifier, ***fitid***
<NOTIFYWILLING>	Flag indicating that client is prepared to send notifications of bill delivery, if desired (see section 14.4.5), ***Boolean***
<INCLUDEDETAIL>	Flag indicating bill detail should be included too, ***Boolean***
</**PRESLISTRQ**>	Closing tag for bill list request

14.4.2.2 Bill List Response <PRESLISTRS>

The <PRESLISTRS> response must appear within a <PRESLISTTRNRS> transaction wrapper.

The <PRESLISTRS> response can contain zero or more bill summaries, with optional detail. Each bill summary corresponds to a (usually monthly) bill.

Tag	*Description*
<**PRESLISTRS**>	Opening tag for bill list response
<BILLPUB>	Official standard name of bill publisher, ***A-32***
<USERID>	User whose bill data is being returned, ***A-32***
<DTSTART>	Start date of bills returned, ***datetime***
<DTEND>	Date to present as start date for next request, ***datetime***
<PRESLIST>	Opening tag for bill summary list
</PRESLIST>	
</**PRESLISTRS**>	Closing tag for bill list response

14.4.2.2.1 Bill List <PRESLIST>

The bill list aggregate <PRESLIST> contains a list of zero or more <PRESBILLINFO> aggregates.

Tag	*Description*
<**PRESLIST**>	Opening tag for bill list
<PRESBILLINFO>	Bill information aggregate (zero or more)
</PRESBILLINFO>	
</**PRESLIST**>	Closing tag for bill list

14.4.2.2.2 Bill Information <PRESBILLINFO>

The bill information aggregate <PRESBILLINFO> provides information about a single bill, including the amount due, date due, and pointers to more information.

If the client requested bill detail in the <PRESLISTRQ>, the bill publisher provides the detail in zero or more <PRESDETAILRS> aggregates. If the client did not request bill detail, the server should use the <DETAILAVAILABLE> flag to indicate whether the client can request bill detail at a later time using the <PRESDETAILRQ> aggregate.

The bill identifier <BILLID> must uniquely identify the bill with the bill publisher (not merely with the biller).

The bill date <DTBILL> is usually a fixed number of days after the end of the bill period. It is not the date on which the bill publisher received the bill for publication.

Tag	*Version*	*Description*
<**PRESBILLINFO**>		Opening tag for bill information
<BILLID>		Identifier for this bill within the bill publisher, ***fitid***
<PRESACCTFROM>		Biller account information (see section 14.3.2.2)
</PRESACCTFROM>		
<PAYEEID2>		Payee identifier. Specify only if the bill publisher is also provides Bill Payment service. See section 14.5. ***SRVRTID2***
<BILLREFINFO>		Biller-defined text to include with the payment, for the biller's Accounts Receivable reconciliation. Sections 14.5, 12.5.2. ***A-80***
<AMTDUE>		Full payment amount due, ***amount***
<MINAMTDUE>		Minimum payment amount due, ***amount***

Tag	Version	Description
<DTPMTDUE>		Payment due date, ***datetime***
<DTBILL>		Bill date, ***datetime***
<DTOPEN>		Opening statement date, ***datetime***
<DTCLOSE>		Closing statement date, ***datetime***
<PREVBAL>		Balance of the account as of the previous period, ***amount***
<ACTIVITY>		Net inflows and outflows for the account since the last period, ***amount***
<ACCTBAL>		Balance of the account at the end of the current period, ***amount***
<INVOICE>		Optional INVOICE tag that the biller would like to receive with a payment (See 12.5.2.2). Client applications should allow the user to edit the amounts before returning this in a payment ***Note: The <LITMCODE> supplied by the biller should not be modified.***
</INVOICE>		
<NOTIFYDESIRED>		Indicator that a delivery notification (see section 14.4.5) is desired, ***Boolean***
<STMNTIMAGE>		Statement image aggregate, see section 14.4.2.2.3
</STMNTIMAGE>		
Choose DETAILAVAILABLE or PRESDETAIL, but not both		
<DETAILAVAILABLE>		Indicator that structured detail is available, ***Boolean***
-or-		
<PRESDETAIL>		Bill details, when requested, see section 14.4.3.2
</PRESDETAIL>		
</PRESBILLINFO>		Closing tag for bill information

If <PREVBAL>, <ACTIVITY>, and <ACCTBAL> are all present, then <PREVBAL> and <ACTIVITY> must add up to <ACCTBAL>.

Note: *this means payments from the consumer received by the biller are counted as negative activity.*

14.4.2.2.3 Statement Image <STMNTIMAGE>

The <STMNTIMAGE> aggregate provides one or more URLs that point to a fully rendered image of the bill, in HTML.

<IMAGEURL> accesses the complete bill image. This URL may contain navigation to other sites, or to other pages of bill images at the same site.

To support off-line viewing of the bill, the server may provide one or more additional URLs. Each <PREFETCHURL> points to a local Web page.

Each URL associated with <IMAGEURL> and <PREFETCHURL> must include an authentication token at the end (for example, ?authtoken=*randomString*). These embedded tokens guarantee that only the customer can access the Web page. Accessing the statement image requires SSL. The bill publisher might include an expiration date for the authentication token, and hence for the URLs. The expiration date could be quite short (for example, 1 hour) or quite long (for example, 1 month). After the expiration date, the client can obtain a new authentication token only by sending a new <PRESLISTRQ> request.

Tag	Description
<STMNTIMAGE>	Opening tag for statement image
<IMAGEURL>	URL address for retrieving an image of the complete bill encoded as

Tag	Description
	HTML. This can be cached by the client for later display, or it can be viewed live directly from the Web. ***URL2***
<PREFETCHURL>	Advice in support of off-line viewing. Zero or more. ***URL2***
<DTEXPIRE>	Date after which embedded authentication token expires, ***datetime***
</**STMNTIMAGE**>	Closing tag for statement image

14.4.2.3 Bill List Transaction Request <PRESLISTTRNRQ>

As the transaction wrapper for <PRESLISTRQ>, this aggregate specifies whether the client is requesting bills for a single user or a group of users. The optional <USERID> and <GROUPID> elements support the following scenarios:

- **A customer requests his or her own bills from the bill publisher:** In this case, the client can optionally specify the customer's <USERID> in the <PRESLISTTRNRQ>. If the client does not specify <USERID>, the bill publisher uses the <USERID> in the signon request <SONRQ>.
- **A customer service representative requests a bill on behalf of a user:** The client sends the representative's <USERID> in the signon request <SONRQ>. To specify the user for which the representative is retrieving bills, the client sends the customer's <USERID> in the <PRESLISTTRNRQ>. The bill publisher must ultimately decide whether the customer service representative can access the requested bills. In its response, the bill publisher includes only those bills for which the requester has access privileges.
- **A client proxy system fetches bills on behalf of a group of users:** Instead of sending a <USERID>, the client sends the <GROUPID> that identifies the group of users. Within the <PRESLISTTRNRS>, the bill publisher returns a <PRESLISTRS> for each user in the named group. Again, the bill publisher decides whether access should be granted. Bill publishers that support usage of the <GROUPID> must maintain knowledge of which users are in which named group. The Open Financial Exchange specification does not provide a way to track membership in a named group. Any such management must happen out-of-band.

Tag	Description
<**PRESLISTTRNRQ**>	Opening tag for bill list transaction request
<TRNUID>	Client-assigned globally unique ID for this transaction, ***trnuid***
<CLTCOOKIE>	Data to be echoed in the transaction response, ***A-32***
<TAN>	Transaction authorization number, ***A-80***
<USERID>	If present, the bill request is on behalf of this particular user, ***A-32***
- or -	
<GROUPID>	If present, the bill request is on behalf of all users in the named group. If both <USERID> and <GROUPID> are absent, the <USERID> in the <SONRQ> is implied. ***A-32***
<PRESLISTRQ>	Bill List Request Aggregate
</PRESLISTRQ>	
</**PRESLISTTRNRQ**>	Closing tag for bill list transaction request

14.4.2.4 Bill List Transaction Response <PRESLISTTRNRS>

The transaction response returns zero or more bills <PRESLISTRS>.

Tag	Description
<**PRESLISTTRNRS**>	Opening tag for bill list transaction response
<TRNUID>	Client-assigned globally unique ID for this transaction ***trnuid***
<STATUS>	Status aggregate

Tag	Description
</STATUS>	
<CLTCOOKIE>	Client provided data. **REQUIRED** if provided in request ***A-32***
<PRESLISTRS>	Bill list response aggregate, zero or more
</PRESLISTRS>	
</**PRESLISTTRNRS**>	Closing tag for bill list transaction response

14.4.2.5 Status Codes <PRESLISTTRNRS>

Code	Meaning
0	Success (INFO)
2000	General error (ERROR)
2002	General account error (ERROR)
2003	Account not found (ERROR)
2004	Account closed (ERROR)
2005	Account not authorized (ERROR)

14.4.3 Bill Detail Retrieval

If statement detail is available for a bill, the client can retrieve the detail using a bill detail request <PRESDETAILRQ>. One example of statement detail is the individual telephone calls from a telephone bill.

14.4.3.1 Bill Detail Request <PRESDETAILRQ>

The <PRESDETAILRQ> request must appear within a <PRESDETAILTRNRQ> transaction wrapper.

Tag	Description
<**PRESDETAILRQ**>	Opening tag for bill detail request
<BILLID>	Statement identifier from <PRESBILLINFO>, ***fitid***
<BILLDETAILTABLETYPE>	If present, filters response to just tables of this type. See table 14.4.3.2.3
</**PRESDETAILRQ**>	Closing tag for bill detail request

14.4.3.2 Bill Detail Response <PRESDETAILRS>

The <PRESDETAILRS> request must appear within a <PRESDETAILTRNRS> transaction wrapper.

The bill detail response contains zero or more <BILLDETAILTABLE> aggregates.

Tag	Description
<**PRESDETAILRS**>	Opening tag for bill detail response
<PRESDETAIL>	Zero or more bill detail aggregates
<BILLID>	Statement identifier from <PRESBILLINFO>, ***fitid***
<PRESACCTFROM>	Identifies biller account. Must be included if in response to an <PRESDETAILRQ>, is redundant inside <PRESBILLINFO>
</PRESACCTFROM>	
<BILLDETAILTABLE>	Zero or more bill detail table aggregates. See section 14.4.3.2.1.
</BILLDETAILTABLE>	
</PRESDETAIL>	Closing tag for bill detail aggregate
</**PRESDETAILRS**>	Closing tag for bill detail response

14.4.3.2.1 Bill Detail Table <BILLDETAILTABLE>

The bill detail table allows billers to send tabular data to the customer in a flexible way. The table might contain phone calls from a telephone bill, or electrical meter readings for a utility bill.

A table consists of one or more rows, each having one or more columns. Within a table, all rows must have identical structures. The <BILLDETAILTABLETYPE> determines the "shape" or schema of the table. The <BILLDETAILTABLENAME> gives a name to this table, and should be unique within an <PRESDETAILRS>.

Tag	*Description*
<**BILLDETAILTABLE**>	Opening tag for bill detail table
<TABLENAME>	Name of bill detail table, ***A-32***
<BILLDETAILTABLETYPE>	Type of bill detail table, see section 14.4.3.2.3
<BILLDETAILROW>	Zero or more bill detail row aggregates, see section 14.4.3.2.2
</BILLDETAILROW>	
</**TABLE**>	Closing tag for bill detail table

14.4.3.2.2 Bill Detail Row <BILLDETAILROW>

A <BILLDETAILTABLE> contains zero or more bill detail rows <BILLDETAILROW>.

A <BILLDETAILROW> contains zero or more columns <C>, whose meanings are specific to the type of table <BILLDETAILTABLETYPE> in which they occur. For the purpose of the DTD parser, all columns <C> are consider to be **Format: A-255**.

Open Financial Exchange requires all elements return data. If bill publishers do not use specific columns, they can return null columns, represented by the element <N>. All columns <N> are considered to be **Format: A-1.**

> ***Note:*** *Bill publishers must include one character of data in a null column. Bill publishers can omit blank columns at the end of a <BILLDETAILROW>, tag and all. (DTD should not enforce ordering, i.e. it should look like this:*

<!ELEMENT BILLDETAILROW - - (C | N)>*

Tag	*Description*
<**BILLDETAILROW**>	Opening tag for bill detail row
<C>	Zero or more column data elements, ***A-255***
<N>	Zero or more column data elements, ***A-1***
</**BILLDETAILROW**>	Closing tag for bill detail row

14.4.3.2.3 Table Types <TABLETYPE>

Open Financial Exchange defines some common table types. Individual billers can define their own table types, and hence their own table structures, but must honor the custom tag naming convention outlined in section 2.7.

Value	*Description*
TransactionList	Table defined for "payment register"-style line items
CallLog	Table defined for record of telephone calls
XXX.Usage	Table defined by biller, not by Open Financial Exchange

14.4.3.2.3.1 TransactionList Table Type

<BILLDETAILTABLE> aggregates marked with <BILLDETAILTABLETYPE> TransactionLists have rows of 14 columns. The first column contains a unique identifier (like a BILLID), and must be present. Other columns may not always apply and can be left blank.

The TransactionList table type is a subset of the <STMTTRN> aggregate in section 11.4.3.

Column	*Name*	*Description*
1	BillId	Unique identifier token from server, ***fitid***
2	TrnType	Transaction type (see section 11.4.3.1)
3	DtPosted	Date item was posted, ***datetime***
4	DtUser	Date user initiated transaction, if known, ***datetime***
5	TrnAmount	Amount of transaction, ***amount***
6	CorrectBillId	If present, unique identifier of previously sent transaction that is corrected by this record, ***fitid***
7	CorrectAction	Replace or delete. Specify only if column 6 is present.
8	CheckNum	Check or other reference number, ***A-12***
9	RefNum	Other reference number, ***A-12***
10	SIC	Standard Industrial Code, ***N-6***
11	Name	Name of payee or description of transaction, ***A-32***
12	Memo	Extra Information, ***A-255***
13	OrigCurSym	Original Currency Identifier (ISO 42173 3-letter), ***A-3***
14	CurRate	Currency rate, ratio of currency to original currency, ***rate***

14.4.3.2.3.2 CallLog Table Type

<BILLDETAILTABLE> aggregates marked with <BILLDETAILTABLETYPE>CallLog have rows of 10 columns.

Column	*Name*	*Description*
1	TNCalledFrom	Telephone number called from, ***A-32***
2	CityStateFrom	City, state (or place, region) called from, ***A-16***
3	TNCalled	Telephone number called, ***A-32***
4	CityState	City, state (or place, region) called, ***A-16***
5	Originated	Date/time call started, ***datetime***
6	Type	Type of call, ***A-8***
7	Rate	Rate (for example, Night, Day, Eve, Wknd), ***A-5***
8	Duration	Duration of call in tenths of seconds, ***N-6***
9	Cost	Cost of call, ***amount***
10	TNChargedTo	Telephone number charged to, ***A-32***

14.4.3.3 Status Codes <PRESDETAILTRNRS>

Code	*Meaning*
0	Success (INFO)
2000	General error (ERROR)
2023	Unknown BILLID (ERROR)
10600	Table type not found (ERROR)

14.4.4 Table Structure Definition

Clients can obtain the definition of a table structure by sending a table structure request <BILLTBLSTRUCTRQ>.

Clients need only request the structure of tables it does not already know about. For instance, the client might request the structure of a biller-specific table that starts with the *x*- prefix. Knowing the structure of a table allows the client to display the data more clearly or store the data in a more compact form, such as a database table.

14.4.4.1 Table Structure Request <TBLSTRUCTRQ>

To identify the table, the client includes the type of table <BILLDETAILTABLETYPE> and unique identifier <BILLID> for the table. Although <BILLDETAILTABLETYPE> uniquely identifies the table, Open Financial Exchange requires the <BILLID> as well to allow various server implementations.

The <BILLTBLSTRUCTRQ> request must appear within a <BILLTBLSTRUCTTRNRQ> transaction wrapper.

Tag	*Description*
<**BILLTBLSTRUCTRQ**>	Opening tag for the table structure request
<BILLID>	Statement Identifier, ***fitid***
<TABLETYPE>	Table type for which the structure is requested
</BILLTBLSTRUCTRQ>	Closing tag for the table structure request

14.4.4.2 Table Structure Response <BILLTBLSTRUCTRS>

The <BILLTBLSTRUCTRS> response must appear within a <BILLTBLSTRUCTTRNRS> transaction wrapper.

The table structure response contains one or more column type definitions, which correspond positionally with the <C> aggregates in a <BILLDETAILROW> in a <BILLDETAILTABLE> of the corresponding <TABLETYPE>.

Tag	*Description*
<**BILLTBLSTRUCTRS**>	Opening tag for table structure response
<BILLID>	Table identifier, ***fitid***
<TABLETYPE>	Table type
<COLDEF>	Zero or more column definition aggregates (see section 14.4.4.2.1)
</COLDEF>	
</BILLTBLSTRUCTRS>	Closing tag for table structure response

14.4.4.2.1 Column Definition <COLDEF>

A column definition <COLDEF> associates a name and a data type with a column.

Tag	*Description*
<**COLDEF**>	Opening tag for column definition
<COLNAME>	Column name, ***A-32***
<COLTYPE>	Column type, in A-255/D/N-6 notation, ***A-8***
</COLDEF>	Closing tag for column definition

14.4.4.3 Status Codes <TBLSTRUCTRS>

Code	*Meaning*

Code	Meaning
0	Success (INFO)
2000	General error (ERROR)
2023	Unknown BILLID (ERROR)
10600	Table type not found (ERROR)

14.4.5 Delivery Notification

The bill publisher can request delivery notification through the <NOTIFYDESIRED> flag in the <PRESBILLINFO> aggregate (see section 14.4.2.2.2). The bill publisher will expect to receive the delivery notification only if the <PRESLISTRQ> had the <NOTIFYWILLING> flag set.

The delivery notification request tells the bill publisher that the client has presented the specified bills to the customer. This is a stronger statement than acknowledging that the bills have been received by the client, specifically when the client software implements the pre-fetching or (push) model.

However, Open Financial Exchange does not define the meaning of "presenting to the customer." In particular, receipt of a delivery notification by the bill publisher has no legal significance. Open Financial Exchange also does not define the maximum elapsed time between the presentation of the bill and the delivery notification.

14.4.5.1 Delivery Notification Request <PRESNOTIFYRQ>

The <PRESNOTIFYRQ> request must appear within a <PRESNOTIFYTRNRQ> transaction wrapper.

Tag	Description
<**PRESNOTIFYRQ**>	Opening tag for delivery notification request
<PRESDELIVERYID>	A bill delivery ID aggregate (see section 14.4.5.1.1)
</PRESDELIVERYID>	
</PRESNOTIFYRQ>	Closing tag for delivery notification Request

14.4.5.1.1 Bill Delivery Identification <PRESDELIVERYID>

This aggregate identifies a bill delivery instance and suggests when the bill was "seen."

<DTSEEN> is the date and time at which the client displayed the bill to the customer. There is no legal significance to this bill delivery identification.

Tag	Description
<**PRESDELIVERYID**>	Opening tag for the bill delivery identification
<PRESACCTFROM>	Biller account information
</PRESACCTFROM>	
<BILLID>	Identifies the bill from the given biller, ***fitid***
<DTSEEN>	Date and time at which the bill was made available to the requester's client, ***datetime***
</PRESDELIVERYID>	Closing tag for the bill delivery identification

14.4.5.2 Delivery Notification Response <PRESNOTIFYRS>

The <PRESNOTIFYRS> response must appear within a <PRESNOTIFYTRNRS> transaction wrapper.

The delivery notification response lets the client know that the delivery notification request was received by the bill publisher.

Tag	*Description*
<**PRESNOTIFYRS**>	Opening tag for Delivery Notification Response
<PRESDELIVERYID>	Zero or more bill delivery ID aggregates
</PRESDELIVERYID>	
</PRESNOTIFYRS>	Closing tag for Delivery Notification Response

14.4.5.3 Status Codes <PRESNOTIFYTRNRS>

Code	*Meaning*
0	Success (INFO)
2000	General error (ERROR)
2023	BILLID not found (ERROR)

14.5 Bill Payment

To pay a bill received through a <PRESLISTRQ> request, the client can use the Bill Payment message set defined in Chapter 12, "Payments." To construct the payment information <PMTINFO> (see section 12.5.2), the client can use the bill information from <PRESBILLINFO>.

14.5.1 Remittance Information

The client should include the <BILLREFINFO> from the <PRESBILLINFO> aggregate as the <BILLREFINFO> in the <PMTINFO> aggregate. This token allows the biller to link the payment with the bill.

14.5.2 Payee Identification

Client software can produce <PAYEEID> or <PAYEE> in one of two ways.

If the same company provides Bill Presentment and Bill Payment services, the client can use the <PAYEEID> included in the <PRESBILLINFO> aggregate.

If the Bill Payment provider is a different company, the client must use information from the <PRESACCTINFO> to construct the <PAYEE> information.

14.6 Bill Presentment E-Mail

Open Financial Exchange currently defines a Bill Presentment e-mail message that clients can send to bill publishers. With this message, a customer can send a message to a bill publisher regarding one of his or her accounts.

The server acknowledges receipt of the message. The bill publisher then prepares a response that the client picks up when it synchronizes with the server. E-mail is subject to synchronization, using <PRESMAILSYNCRQ> and <PRESMAILSYNCRS>.

Client Sends	*Server Responds*
Addressed message	
PRES account information	
	Acknowledgment

Client Sends	Server Responds
. . . Synchronization request	 Response to customer

14.6.1 Bill Presentment Mail Request <PRESMAILRQ>

The client must identify the account to which account the customer query is related.

The <PRESMAILRQ> request must appear with a <PRESMAILTRNRQ> transaction wrapper.

Tag	Description
<PRESMAILRQ>	PRES-e-mail-request aggregate
<PRESACCTFROM>	Account-from aggregate
</PRESACCTFROM>	
<MAIL>	To, from, message information, see section 9.2.2
</MAIL>	
</**PRESMAILRQ**>	

14.6.2 Bill Presentment Mail Response <PRESMAILRS>

The <PRESMAILRQ> request must appear with a <PRESMAILTRNRQ> transaction wrapper.

Tag	Description
<PRESMAILRS>	PRES-e-mail-response aggregate
<PRESACCTFROM>	Account-from aggregate
</PRESACCTFROM>	
<MAIL>	To, from, message information, see section 9.2.2
</MAIL>	
</**PRESMAILRS**>	

14.6.3 Status Codes <PRESMAILTRNRS>

Code	Meaning
0	Success (INFO)
2000	General error (ERROR)
2002	General account error (ERROR)
2003	Account not found (ERROR)
2004	Account closed (ERROR)
2005	Account not authorized (ERROR)
2019	Duplicate request (ERROR)
16500	HTML not allowed (ERROR)
16501	Unknown mail To: (ERROR)

14.7 Message Sets and Profile

Open Financial Exchange separates the messages that the client and server send into groups called message sets. In its profile response <PROFRS>, each bill publisher or other server provider defines the message sets that it supports and any options available for those message sets.

This section defines the message sets supported by Bill Presentment. It then describes the corresponding message set profile aggregates that can be provided in the profile response <PROFRS>. The message set profile aggregates for the <PROFRS> allow a bill publisher or other server provider to customize its use of Open Financial Exchange. For example, a server might support the Bill Delivery message set <PRESDLVMSGSET>, but not the Group Account Information message set <PRESDIRMSGSETV1>.

For general information about profiles, see Chapter 7, "FI Profile."

14.7.1 Message Sets and Messages

Bill Presentment defines the following message sets:

- Biller Directory message set <PRESDIRMSGSET>, which includes messages for finding billers and bill publishers
- Bill Delivery message set <PRESDLVMSGSET>, which includes messages for delivering bills and bill detail to customers, as well as messages for getting account information for a group of users
-

14.7.1.1 Biller Directory Message Set and Messages

14.7.1.1.1 Biller Directory Request Messages

Message Set	*Message*
<PRESDIRMSGSET>	
<PRESDIRMSGSETV1>	
<PRESDIRMSGSRQV1>	FINDBILLERTRNRQ FINDBILLERRQ
</PRESDIRMSGSRQV1>	
</PRESDIRMSGSETV1>	
</PRESDIRMSGSET>	

14.7.1.1.2 Biller Directory Response Messages

Message Set	*Message*
<PRESDIRMSGSET>	
<PRESDIRMSGSETV1>	
<PRESDIRMSGSRSV1>	FINDBILLERTRNRS FINDBILLERRS
</PRESDIRMSGSRSV1>	
</PRESDIRMSGSETV1>	
</PRESDIRMSGSET>	

14.7.1.2 Bill Delivery Message Set and Messages

14.7.1.2.1 Bill Delivery Request Messages

Message Set	*Message*
<PRESDLVMSGSET>	
<PRESDLVMSGSETV1>	
<PRESDLVMSGSRQV1>	PRESLISTTRNRQ
	PRESLISTRQ
	PRESDETAILTRNRQ
	PRESDETAILRQ
	BILLTBLSTRUCTTRNRQ
	BILLTBLSTRUCTRQ
	PRESNOTIFYTRNRQ
	PRESNOTIFYRQ
	PRESGRPACCTINFOTRNRQ
	ACCTINFORQ
	PRESMAILTRNRQ
	PRESMAILRQ
	PRESMAILSYNCRQ
</PRESDLVMSGSRQV1>	
</PRESDLVMSGSETV1>	
</PRESDLVMSGSET>	

14.7.1.2.2 Bill Delivery Response Messages

Message Set	*Message*
<PRESDLVMSGSET>	
<PRESDLVMSGSETV1>	
<PRESDLVMSGSRSV1>	PRESLISTTRNRS
	PRESLISTRS
	PRESDETAILTRNRS
	PRESDETAILRS
	BILLTBLSTRUCTTRNRS
	BILLTBLSTRUCTRS
	PRESNOTIFYTRNRS
	PRESNOTIFYRS
	PRESGRPACCTINFOTRNRS
	ACCTINFORS
	PRESMAILTRNRS
	PRESMAILRS
	PRESMAILSYNCRS
</PRESDLVMSGSRSV1>	

Message Set	Message
</PRESDLVMSGSETV1>	
</PRESDLVMSGSET>	

14.7.2 Biller Directory Message Set Profile

This section defines the profile aggregate for the Biller Directory message set. This profile aggregate should be included in the <PROFRS> response for those servers that support the Biller Directory message set.

Message Set	Message
<PRESDIRMSGSET>	Opening tag for the Biller Directory message set profile
<PRESDIRMSGSETV1>	Version 1 of Biller Directory message set
<MSGSETCORE>	Common message-set core
</MSGSETCORE>	
<PRESDIRPROF>	Directory profile (if supported)
<PROCDAYSOFF>	Days of week that no processing occurs: MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, SATURDAY, or SUNDAY. 0 or more <PROCDAYSOFF> can be sent.
<CANSUPPORTIMAGES>	Supports delivery of images as multi-part MIME, ***Boolean***
<PROCENDTM>	Time of day that day's processing ends, ***time***
</PRESDIRPROF>	
</PRESDIRMSGSETV1>	
</PRESDIRMSGSET>	Closing tag for the Biller Directory message set profile

14.7.3 Bill Delivery Message Set Profile

This section defines the profile aggregate for the Bill Delivery message set. This profile aggregate should be included in the <PROFRS> response for those servers that support the Bill Delivery message set.

Tag	Description
<PRESDLVMSGSET>	Opening tag for the Bill Delivery message set profile
<PRESDLVMSGSETV1>	Version 1 of Bill Delivery message set
<MSGSETCORE>	Common message-set core
</MSGSETCORE>	
<PRESDLVPROF>	Bill Delivery profile (if supported)
<GROUPID>	Supports account information requests for a group of users, that is PRESGRPACCTINFOTRNRQ, ***Boolean***
<PROCDAYSOFF>	Days of week that no processing occurs: MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, SATURDAY, or SUNDAY. 0 or more <PROCDAYSOFF> can be sent.
<CANSUPPORTIMAGES>	Supports delivery of images as multi-part MIME, ***Boolean***
<PROCENDTM>	Time of day that day's processing ends, ***time***
</PRESDLVPROF>	
<EMAILPROF>	E-mail profile
<CANEMAIL>	Supports generalized e-mail, ***Boolean***
<CANNOTIFY>	Supports notification (of any kind), ***Boolean***
</EMAILPROF>	
</PRESDLVMSGSETV1>	

Tag	Description
</PRESDLVMSGSET>	Closing tag for the Bill Delivery message set profile

14.8 Bill Presentment Examples

14.8.1 Find Biller Examples

14.8.1.1 Get All Billers

The client sends a <FINDBILLERRQ> request to retrieve all available billers:

```
<FINDBILLERRQ>      <!--Beginning of find biller request-->
   <INCIMAGES>N                            <!--LOGO Images are not requested-->
</FINDBILLERRQ>                            <!--End of request-->
```

To keep the size of the example reasonable, we will assume that there are only four billers. Here is the server reply.

```
<FINDBILLERRS>      <!--Beginning of response-->
   <DTUPDATE>19960415092000                <!--Date last update 04/15/97 9:20am-->
   <BILLERINFO>
      <BILLPUB>Wepubbills                  <!--Name of Bill Publisher-->
      <BILLERID>123456789                  <!--Biller ID at Wepubbills-->
      <NAME>RealBig Credit Co.             <!--Name of biller-->
      <ADDR1>1324 Whatever St.             <!--Street address of biller-->
      <CITY>MajorMetro                     <!--City of the Biller-->
      <STATE>OH                            <!--State of the biller-->
      <POSTALCODE>12345-1234               <!--Postal code of biller-->
      <COUNTRY>USA
      <SIC>23                              <!--Standard Industry Code of biller-->
      <PHONE>614-235-2323                  <!--Biller's phone number-->
      <PAYMENTINSTRUMENTS>                 <!--Type of payment accepted-->
         <PAYMENTINSTRUMENT>
            <PMTINSTRUMENTTYPE>CHECKINGACCOUNT
         </PAYMENTINSTRUMENT>
         <PAYMENTINSTRUMENT>
            <PMTINSTRUMENTTYPE>CONCENTRATOR
            <BRAND>CityBank
         </PAYMENTINSTRUMENT>
      </PAYMENTINSTRUMENTS>
      <ACCTFORMAT>([0-9]\{3\}-)\{3\}       <!--Regular expression describing -->
                                           <!—-biller's account number-->
      <ACCTEDITMASK>"    -    -    -    "
                                           <!--Edit mask for account number-->
      <LOGO>http://www.realbig.com/logo.gif
                                           <!--URL to logo of biller-->
   </BILLERINFO>
   <BILLERINFO>
      <BILLPUB>Wepubbills                  <!--Name of Bill Publisher-->
      <BILLERID>222334465                  <!--Biller ID at Wepubbills-->
      <NAME>Aphone Company                 <!--Name of biller-->
      <ADDR1>1324 Where Blvd               <!--Street address of biller-->
      <CITY>Sometown<!--City of the biller-->
      <STATE>CA                            <!--State of the biller-->
      <POSTALCODE>10992-1234               <!--Postal code of biller-->
      <COUNTRY>USA
      <SIC>39                              <!--Standard Industry Code of biller-->
      <PHONE>345-345-3489                  <!--Biller's phone number-->
      <PAYMENTINSTRUMENTS>                 <!--Type of payment accepted-->
         <PAYMENTINSTRUMENT>
            <PMTINSTRUMENTTYPE>CHECKINGACCOUNT
```

```
         </PAYMENTINSTRUMENT>
      </PAYMENTINSTRUMENTS>
      <ACCTFORMAT>([1-9]\{2\}-)\{2\}[0-9]\{3\}
                                          <!--Regular expression describing-->
                                          <!--biller's account number-->
      <ACCTEDITMASK>"   -   -    "        <!--Edit mask for account number-->
      <LOGO>http://www.webup.com/aphone.gif
                                          <!--URL to logo of biller-->
   </BILLERINFO>
   <BILLERINFO>
      <BILLPUB>Wepubbills                 <!--Name of Bill Publisher-->
      <BILLERID>98765123454               <!--Biller ID at Wepubbills-->
      <NAME>Goodol Mortgage               <!--Name of biller-->
      <ADDR1>8273 Magnolia St.            <!--Street address of biller-->
      <CITY>Atlanta                       <!--City of the Biller-->
      <STATE>GA                           <!--State of the biller-->
      <POSTALCODE>34342-6789              <!--Postal code of biller-->
      <COUNTRY>USA
      <SIC>03                             <!--Standard Industry Code of biller-->
      <PHONE>864-234-6745                 <!--Biller's phone number-->
      <PAYMENTINSTRUMENTS>                <!--Type of payment accepted-->
         <PAYMENTINSTRUMENT>
            <PMTINSTRUMENTTYPE>CHECKINGACCOUNT
         </PAYMENTINSTRUMENT>
      </PAYMENTINSTRUMENTS>
      <ACCTFORMAT>[0-1]\{12\}             <!--Regular expression describing-->
                                          <!--biller's account number-->
      <HELPMESSAGE>Enter the first 13 digits of your account number
                                          <!—-to help user key account number-->
      <RESTRICT> GA residents only.       <!--Indicate restricted availibility-->
      <LOGO>http://www.wepub.com/mort.gif
                                          <!--URL to logo of biller-->
   </BILLERINFO>
   <BILLERINFO>
      <BILLPUB>Wepubbills                 <!--Name of Bill Publisher-->
      <BILLERID>32812816734               <!--Biller ID at Wepubbills-->
      <NAME>Sam's Widgets                 <!--Name of biller-->
      <ADDR1>Apt B3                       <!--Street address of biller-->
      <ADDR2>1267 Tank Rd
      <CITY>Columbus<!--City of Biller-->
      <STATE>OH                           <!--State of the biller-->
      <POSTALCODE>77723-8989              <!--Postal code of biller-->
      <COUNTRY>USA
      <SIC>12                             <!--Standard Industry Code of biller-->
      <PHONE>614-657-8934                 <!--Biller's phone number-->
      <PAYMENTINSTRUMENTS>                <!--Type of payment accepted-->
         <PAYMENTINSTRUMENT>
            <PMTINSTRUMENTTYPE>CHECKINGACCOUNT
         </PAYMENTINSTRUMENT>
         <PAYMENTINSTRUMENT>
            <PMTINSTRUMENTTYPE>CONCENTRATOR
            <BRAND>BigConcentrator
         </PAYMENTINSTRUMENT>
      </PAYMENTINSTRUMENTS>
      <ACCTEDITMASK>"   -   - "           <!--Edit mask for account number-->
      <LOGO>http://www.relbig.com/logo.gif
                                          <!--URL to logo of biller-->
      <VALIDATE>http://www.wepub.com/sam.cgi
                                          <!--URL used to validate acct number-->
   </BILLERINFO>
</FINDBILLERRS>
```

14.8.1.2 Find Selected Billers

In this example, the client requests only those billers that are located in Ohio:

```
<FINDBILLERRQ>
   <STATE>OH
   <INCIMAGES>N
</FINDBILLERRQ>
```

In the same circumstances as before, the response would be:

```
<FINDBILLERRS>
   <DTUPDATE>19960415092000                <!--Date last update 04/15/97 9:20am-->
   <BILLERINFO>
      <BILLPUB>Wepubbills                  <!--Name of Bill Publisher-->
      <BILLERID>123456789                  <!--Biller ID at Wepubbills-->
      <NAME>RealBig Credit Co.             <!--Name of biller-->
      <ADDR1>1324 Whatever St.             <!--Street address of biller-->
      <CITY>MajorMetro                     <!--City of the Biller-->
      <STATE>OH                            <!--State of the biller-->
      <POSTALCODE>12345-1234               <!--Postal code of biller-->
      <COUNTRY>USA
      <SIC>23                              <!--Standard Industry Code of biller-->
      <PHONE>614-235-2323                  <!--Biller's phone number-->
      <PAYMENTINSTRUMENTS>                 <!--Type of payment accepted-->
         <PAYMENTINSTRUMENT>
            <PMTINSTRUMENTTYPE>CHECKINGACCOUNT
         </PAYMENTINSTRUMENT>
         <PAYMENTINSTRUMENT>
            <PMTINSTRUMENTTYPE>CONCENTRATOR
            <BRAND>CityBank
         </PAYMENTINSTRUMENT>
      </PAYMENTINSTRUMENTS>
      <ACCTFORMAT>([0-1]\{3\}-)\{3\}       <!--Regular expression describing-->
                                           <!--biller's account number-->
      <ACCTEDITMASK>"    -    -    -    "
                                           <!--Edit mask for account number-->
      <LOGO>http://www.relbig.com/logo.gif
                                           <!--URL to logo of biller-->
   </BILLERINFO>
</FINDBILLERRS>
```

14.8.2 Enrollment Examples

In this example, the client wants to enroll with a bill publisher.

14.8.2.1 Enrollment Request Example

```
<OFX>
   <SIGNONMSGSRQV2> <!--Signon Request-->
      <SONRQ>
         <DTCLIENT>19970307022243          <!--Timestamp, 3/07/97, 2:22:43am-->
         <USERID>D                         <!--Dummy userid-->
         <USERPASS>D <!—-Dummy password-->
         <LANGUAGE>ENG
         <APPID>OFXAPP
         <APPVER>0201
      </SONRQ>
   </SIGNONMSGSRQV2>
   <SIGNUPMSGSRQV2> <!--Enrollment Request-->
      <ENROLLTRNRQ>
         <TRNUID>10001
         <ENROLLRQ>
```

```
            <FIRSTNAME>Cindy
            <MIDDLENAME>P
            <LASTNAME>Williams
            <ADDR1>123 Oak St
            <CITY>San Jose
            <STATE>CA
            <POSTALCODE>94111
            <COUNTRY>USA
            <DAYPHONE>415-555-0123
            <EVEPHONE>408-555-2323
            <EMAIL>cindy@aol.com
            <USERID>cindyid
            <TAXID>111-33-5555
            <SECURITYNAME>wynona
            <DATEBIRTH>19650402
         </ENROLLRQ>
      </ENROLLTRNRQ>
   </SIGNUPMSGSRQV2>
</OFX>
```

14.8.2.2 Enrollment Response Example

For this example, the server responds with immediate acceptance. In practice, many servers would send an enrollment status code of 13000 and send the user ID and password in a welcome letter.

```
<OFX>
   <SIGNONMSGSRSV2> <!--Signon response->
      <SONRS>
         <STATUS>
            <CODE>0
            <SEVERITY>INFO
         </STATUS>
         <DTSERVER>19970307081437          <!--Timestamp, 3/07/97, 8:14:37am-->
         <LANGUAGE>ENG
         <DTPROFUP>19970301070000          <!--Timestamp, 3/01/97, 7:00:00am-->
         <DTACCTUP>19970301070000          <!--Timestamp, 3/01/97, 7:00:00am-->
      </SONRS>
   </SIGNONMSGSRSV2>
   <SIGNUPMSGSRSV2> <!--Enrollment response-->
      <ENROLLTRNRS>
         <TRNUID>10001
         <STATUS>
            <CODE>0
            <SEVERITY>INFO
         </STATUS>
         <ENROLLRS>
            <TEMPPASS>y12345
            <USERID>cindyid
            <DTEXPIRE>19970407             <!--When Temp Password Expires-->
         </ENROLLRS>
      </ENROLLTRNRS>
   </SIGNUPMSGSRSV2>
</OFX>
```

14.8.3 Activation Example

After enrollment, Cindy wants to sign up with a biller, presumably found with the directory services, with biller ID 415-552-9923 of bill publisher Publisher, Inc.

14.8.3.1 Activation Request Example

```
<OFX>
```

```
   <SIGNONMSGSRQV2> <!--Signon Request-->
      <SONRQ>
         <DTCLIENT>19970308122243          <!--Timestamp, 3/08/97,12:22:43pm-->
         <USERID>cindyid                   <!—-client’s userid-->
         <USERPASS>y12345                  <!—-client’s temporary password-->
         <LANGUAGE>ENG
         <APPID>OFXAPP
         <APPVER>0201
      </SONRQ>
   </SIGNONMSGSRQV2>
   <SIGNUPMSGSRQV2> <!--Activation Request-->
      <ACCTTRNRQ>
         <TRNUID>10002
         <ACCTRQ>   <!—-Activate biller acct -->
            <SVCADD>
               <PRESACCTTO>
                  <BILLPUB>Publisher, Inc.
                  <BILLER>415-552-9923
                  <ACCTID>4128 9343 2324 2314
                <PRESNAMEADDRESS>
                  <NAME>Cindy P Williams    <!--Name as on biller’s statement-->
                  <ADDR1>123 Oak St         <!--Address as on statement-->
                  <CITY>San Jose
                  <STATE>CA
                  <POSTALCODE>94111
                  <COUNTRY>USA
                  <PHONE>408-555-2323
                </PRESNAMEADDRESS>
                <USERID>cindyid
               </PRESACCTTO>
            </SVCADD>
            <SVC>PRESSVC
         </ACCTRQ>
      </ACCTTRNRQ>
   </SIGNUPMSGSRQV2>
</OFX>
```

14.8.3.2 Activation Response Example

```
<OFX>
   <SIGNONMSGSRSV2> <!--Signon Response->
      <SONRS>
         <STATUS>
            <CODE>0
            <SEVERITY>INFO
         </STATUS>
         <DTSERVER>19970308181437          <!--Timestamp, 3/07/97, 6:14:37pm-->
         <LANGUAGE>ENG
         <DTPROFUP>19970301070000          <!--Timestamp, 3/01/97, 7:00:00am-->
         <DTACCTUP>19970301070000          <!--Timestamp, 3/01/97, 7:00:00am-->
      </SONRS>
   </SIGNONMSGSRSV2>
   <SIGNUPMSGSRSV2> <!--Enrollment Response-->
      <ACCTTRNRS>   <!--Service Activation Response-->
         <TRNUID>10002
         <STATUS>
            <CODE>0
            <SEVERITY>INFO
         </STATUS>
         <ACCTRS>
            <SVCADD>
               <PRESACCTTO>
                  <BILLPUB>Publisher, Inc.
                  <BILLER>415-552-9923
```

```
                <ACCTID>4128 9343 2324 2314
               <PRESNAMEADDRESS>
                <NAME>Cindy P Williams
                <ADDR1>123 Oak St
                <CITY>San Jose
                <STATE>CA
                <POSTALCODE>94111
                <COUTNRY>USA
                <PHONE>408-555-2323
               </PRESNAMEADDRESS>
               <USERID>cindyid
             </PRESACCTTO>
          </SVCADD>
          <SVC>PRESSVC
       </ACCTRS>
    </ACCTTRNRS>
 </SIGNUPMSGSRSV2>
</OFX>
```

14.8.4 Bill Delivery Examples

14.8.4.1 Customer Bill Delivery Example

The customer, Dan North, wants to see his bills since 3/1/97, which is the last time he asked to see his bills.

14.8.4.1.1 Customer Bill Delivery Request Example

```
<OFX>
   <SIGNONMSGSRQV2>
      <SONRQ>
         <DTCLIENT>19970409090000          <!--Current date 4/9/1997-->
         <USERID>123-45-6789               <!--Dan North’s user id-->
         <USERPASS>DansPassword
         <LANGUAGE>ENG
         <APPID>EndUserApp
         <APPVER>0700
      </SONRQ>
   </SIGNONMSGSRQV2>
   <PRESDLVMSGSRQV1>
      <PRESLISTTRNRQ>
         <TRNUID>12345
         <PRESLISTRQ>
            <BILLPUB> ABillPublisher
            <DTSTART>19970301000000        <!--Get Dan's bills since 3/1/97-->
            <INCLUDEDETAIL>Y
         </PRESLISTRQ>
      </PRESLISTTRNRQ>
   </PRESDLVMSGSRQV1>
</OFX>
```

14.8.4.1.2 Customer Bill Delivery Response Example

```
<OFX>
   <SIGNONMSGSRSV2>
      <SONRS>
         <STATUS>
            <CODE>0
            <SEVERITY>INFO
         </STATUS>
         <DTSERVER>19970409001600
      </SONRS>
```

```
   </SIGNONMSGSRSV2>
   <PRESDLVMSGSRSV1>
      <PRESLISTTRNRS>
         <TRNUID>12345
         <STATUS>
            <CODE>0
            <SEVERITY>INFO
         </STATUS>
         <PRESLISTRS>
            <BILLPUB>ABillPublisher
            <USERID>123-45-6789
            <DTSTART>19970301000000         <!--Same as in request: no data loss-->
            <DTEND>19970409090000           <!--Value for DTSTART next time-->
            <PRESLIST>
               <PRESBILLINFO>
                  <BILLID>65432
                  <PRESACCTFROM>
                     <BILLPUB>ABillPublisher
                     <BILLERID>1001         <!--Biller id for Power Inc-->
                     <ACCTID>1245678GL7     <!--Dan North's acct with Power Inc-->
                  </PRESACCTFROM>
                  <BILLREFINFO>1234678GL7970501
                  <AMTDUE>124.24            <!--Dan North to pay $124.24-->
                  <DTPMTDUE>19970501        <!--by 5/1/97 -->
                  <DTBILL>19970401
                  <NOTIFYDESIRED>N
                  <STMNTIMAGE>
                     <IMAGEURL>
https://www.Power.com/bills/apr/dannorth.htm?authtoken=65j3ltfm7
                     <DTEXPIRE>199704101200
                                            <!--Must visit url by 4/10/97 12am-->
                  </STMNTIMAGE>
                  <PRESDETAILRS>
                     <BILLID>65432          <!--This links detail to billinfo-->
                     <BILLDETAILTABLE>
                        <TABLENAME>usage
                        <BILLDETAILTABLETYPE>x_Power_usage
                                            <!--Power Inc format for usage-->
                        <BILLDETAILROW>
                           <C>elec          <!--Consumable-->
                           <C>19970228      <!--Date meter reading start of period->
                           <C>65543         <!--Meter reading at start of period-->
                           <C>19970328      <!--Date meter reading end of period-->
                           <C>65643         <!--Meter reading at end of period-->
                           <C>100           <!--Difference in meter readings-->
                           <C>KWH           <!--Units-->
                           <C>.8934         <!--Rate (price per unit)-->
                           <C>89.34         <!--Charge -->
                        </BILLDETAILROW>
                        <BILLDETAILROW>
                           <C>gas           <!--Consumable -->
                           <C>19970226      <!--Date meter reading start of period->
                           <C>509843        <!--Meter reading at start of period -->
                           <C>19970327      <!--Date meter reading end of period -->
                           <C>510843        <!--Meter reading at end of period -->
                           <C>1000          <!--Difference in meter readings -->
                           <C>Therms        <!--Units -->
                           <C>.02543        <!--Rate (price per unit) -->
                           <C>25.43         <!--Charge -->
                        </BILLDETAILROW>
                     </BILLDETAILTABLE>
                     <BILLDETAILTABLE>
                        <TABLENAME>charges
                        <BILLDETAILTABLETYPE>x-Power-charges
```

```
                        <BILLDETAILROW>
                           <C>75.34         <!--Amount -->
                           <C>Previous Balance 3/1/97
                                            <!--Description -->
                        </BILLDETAILROW>
                        <BILLDETAILROW>
                           <C>75.34
                           <C>Payment Received 3/17/97 - Thank You!
                        </BILLDETAILROW>
                        <BILLDETAILROW>
                           <C>0
                           <C>Open Balance
                        </BILLDETAILROW>
                        <BILLDETAILROW>
                           <C>114.77
                           <C>Services, Subtotal
                        </BILLDETAILROW>
                        <BILLDETAILROW>
                           <C>9.47
                           <C>Taxes, 8.25%
                        </BILLDETAILROW>
                     </BILLDETAILTABLE>
                  </PRESDETAILRS>
               </PRESBILLINFO>
               <PRESBILLINFO>
                  <BILLID>65436
                  <PRESACCTFROM>
                     <BILLPUB>ABillPublisher
                     <BILLERID>2021         <!--Biller id of FluteRental, Inc. -->
                     <ACCTID>8765XY95       <!--Dan North’s account number -->
                  </PRESACCTFROM>
                  <BILLREFINFO>8765XY95970428
                  <AMTDUE>16.21             <!--Total to be paid -->
                  <DTPMTDUE>19970428        <!--by 4/28/97 -->
                  <DTBILL>19970408
                  <NOTIFYDESIRED>N
                  <STMNTIMAGE>
                     <IMAGEURL> https://www.FluteRental.com/95rs3vlx/bill.asp
                     <DTEXPIRE>19970601     <!--Must visit url by 6/1/97 -->
                  </STMNTIMAGE>
                  <DETAILAVAILABLE>N        <!--No structured detail exists -->
               </PRESBILLINFO>
            </PRESLIST>
         </PRESLISTRS>
      </PRESLISTTRNRS>
   </PRESDLVMSGSRSV1>
</OFX>
```

14.8.4.2 Bill Delivery for Customer Service Representative

This example assumes that Dan North calls Power Inc with a question about his power bill. Power's customer service representative, Maria Smith, uses a similar application and a similar Open Financial Exchange request to see the same bill that Dan sees.

14.8.4.2.1 Bill Delivery Request Example for a Customer Service Representative

```
<OFX>
   <SIGNONMSGSRQV2>
      <SONRQ>
         <DTCLIENT>19970410100000
         <USERID>987-65-4321                <!--Maria Smith’s user id -->
```

```
         <USERPASS>MariasPassword
         <LANGUAGE>ENG
         <APPID>CSRApp
         <APPVER>0500
      </SONRQ>
   </SIGNONMSGSRQV2>
   <PRESDLVMSGSRQV1>
      <PRESLISTTRNRQ>
         <TRNUID>23456
         <USERID>123-45-6789                <!--Asks for Dan North’s bills -->
         <PRESLISTRQ>
            <BILLPUB> ABillPublisher
            <DTSTART>19970330               <!--Approximate date -->
            <DTEND>1997040410               <!--Approximate date -->
            <NOTIFYWILLING>N
            <INCLUDEDETAIL>Y
         </PRESLISTRQ>
      </PRESLISTTRNRQ>
   </PRESDLVMSGSRQV1>
</OFX>
```

14.8.4.2.2 Bill Delivery Response Example for a Customer Service Representative

The response from the server includes the Power Incorporated bill, but not the FluteRental bill. This is because the server decides that Maria Smith’s credentials are good enough to see Dan North’s Power Inc bill, but not good enough to see anything else.

```
<OFX>
   <SIGNONMSGSRSV2>
      <SONRS>
         <STATUS>
            <CODE>0
            <SEVERITY>INFO
         </STATUS>
         <DTSERVER>19970410101300
      </SONRS>
   </SIGNONMSGSRSV2>
   <PRESDLVMSGSRSV1>
      <PRESLISTTRNRS>
         <TRNUID>23456
         <STATUS>
            <CODE>0
            <SEVERITY>INFO
         </STATUS>
         <PRESLISTRS>
            <BILLPUB>ABillPublisher
            <USERID>123-45-6789             <!--Dan North’s userid, so Dan’s bill-->
            <DTSTART>19970328               <!--Same as in request: no data loss-->
            <DTEND>19970409                 <!--Same as in request-->
            <PRESLIST>
               <PRESBILLINFO>
                  <BILLID>65432
                  <PRESACCTFROM>
                     <BILLPUB>ABillPublisher
                     <BILLERID>1001         <!--Biller id for Power Inc-->
                     <ACCTID>1245678GL7     <!--Dan's account with Power Inc-->
                     <USERID>123-45-6789    <!--Dan North’s userid-->
                  </PRESACCTFROM>
                  <BILLREFINFO>1234678GL7970501
                  <AMTDUE>124.24            <!--Dan to pay $124.24-->
                  <DTPMTDUE>19970501        <!--by 5/1/97 -->
                  <DTBILL>19970401
                  <NOTIFYDESIRED>N
```

```
                  <STMNTIMAGE>
                     <IMAGEURL>
https://www.Power.com/bills/apr/dannorth.htm?authtoken=987ab6gr8y
                     <DTEXPIRE>199704111200
                                          <!--Must visit url by 4/11/97 12am-->
                  </STMNTIMAGE>
                  <PRESDETAILRS>
                     <BILLID>65432
                     <BILLDETAILTABLE>
                        <TABLENAME>usage
                        <BILLDETAILTABLETYPE>x_Power_usage
                                          <!--Power Inc format for usage-->
                        <BILLDETAILROW>
                           <C>elec        <!--Consumable-->
                           <C>19970228    <!--Date meter reading start of period->
                           <C>65543       <!--Meter reading at start of period-->
                           <C>19970328    <!--Date meter reading end of period-->
                           <C>65643       <!--Meter reading at end of period-->
                           <C>100         <!--Difference in meter readings-->
                           <C>KWH         <!--Units-->
                           <C>.8934       <!--Rate (price per unit)-->
                           <C>89.34       <!--Charge-->
                        </BILLDETAILROW>
                        <BILLDETAILROW>
                           <C>gas         <!--Consumable-->
                           <C>19970226    <!--Date meter reading start of period->
                           <C>509843      <!--Meter reading at start of period-->
                           <C>19970327    <!--Date meter reading end of period-->
                           <C>510843      <!--Meter reading at end of period-->
                           <C>1000        <!--Difference in meter readings-->
                           <C>Therms      <!--Units-->
                           <C>.02543      <!--Rate (price per unit)-->
                           <C>25.43       <!--Charge-->
                        </BILLDETAILROW>
                     </BILLDETAILTABLE>
                     <BILLDETAILTABLE>
                        <TABLENAME>charges
                        <BILLDETAILTABLETYPE>x-Power-charges
                        <BILLDETAILROW>
                           <C>75.34                    <!--Amount-->
                           <C>Previous Balance 3/1/97  <!--Description-->
                        </BILLDETAILROW>
                        <BILLDETAILROW>
                           <C>75.34
                           <C>Payment Received 3/17/97 - Thank You!
                        </BILLDETAILROW>
                        <BILLDETAILROW>
                           <C>0
                           <C>Open Balance
                        </BILLDETAILROW>
                        <BILLDETAILROW>
                           <C>114.77
                           <C>Services, Subtotal
                        </BILLDETAILROW>
                        <BILLDETAILROW>
                           <C>9.47
                           <C>Taxes, 8.25%
                        </BILLDETAILROW>
                     </BILLDETAILTABLE>
                  </PRESDETAILRS>
               </PRESBILLINFO>
            </PRESLIST>
         </PRESLISTRS>
      </PRESLISTTRNRS>
```

```
   </PRESDLVMSGSRSV1>
</OFX>
```

14.8.4.3 Bill Delivery for a Group of Users

In this example, Realtors Company downloads the phone bills for the employees' office phones by asking the bill publisher to see the bills for the group RealtorsEmployees. The composition of the group RealtorsEmployees has been agreed upon between Realtors Company and the bill publisher; moreover, the bill publisher has agreed to grant Realtors Company access rights to the RealtorsEmployees group. All this took place outside of Open Financial Exchange.

14.8.4.3.1 Bill Delivery Request Example for a Group of Users

```
<OFX>
   <SIGNONMSGSRQV2>
      <SONRQ>
         <DTCLIENT>19970515100000
         <USERID>888-66-4444              <!--Realtors Company's user id -->
         <USERPASS>RealtorsPassword
         <LANGUAGE>ENG
         <APPID>CSRApp
         <APPVER>0500
      </SONRQ>
   </SIGNONMSGSRQV2>
   <PRESDLVMSGSRQV1>
      <PRESLISTTRNRQ>
         <TRNUID>34567
         <GROUPID>RealtorsEmployees       <!--Asks for Employee's phone bills-->
         <PRESLISTRQ>
            <BILLPUB> ABillPublisher
            <DTSTART>19970430             <!--since 4/30/1997-->
            <NOTIFYWILLING>N
            <INCLUDEDETAIL>Y
         </PRESLISTRQ>
      </PRESLISTTRNRQ>
   </PRESDLVMSGSRQV1>
</OFX>
```

14.8.4.3.2 Bill Delivery Response Example for a Group of Users

The response, not shown here, will include several bills each marked with its own <USERID>. The only bills returned will be the employees' phone bills for their office phones, since those bills are the only ones to which Realtor Company has access rights.

14.8.4.4 Group Account Information Example

This is an example of a client proxy system that is tracking changes to the accounts of a group of users.

14.8.4.4.1 Group Account Information Request Example

```
<OFX>
   <SIGNONMSGSRQV2> <!--Signon Request->
      <SONRQ>
         <DTCLIENT>19970307022243         <!--Timestamp, 3/07/97, 2:22:43am-->
         <USERID>AClientProxySystem
         <USERPASS>T23qdfw2               <!--Password of proxy system-->
         <LANGUAGE>ENG
         <APPID>OFXAPP
         <APPVER>0201
```

```
      </SONRQ>
   </SIGNONMSGSRQV2>
   <PRESDLVMSGSRQV1><!--Group Account Info Request-->
      <PRESGRPACCTINFOTRNRQ>
         <TRNUID>10001
         <GROUPID>AClientProxysCustomers
                                          <!--Predefined group of customers-->
         <ACCTINFORQ>
            <DTACCTUP>19970104            <!--Last DTACCTUP received for group-->
         </ACCTINFORQ>
      </PRESGRPACCTINFOTRNRQ>
   </PRESDLVMSGSRQV1>
</OFX>
```

14.8.4.4.2 Group Account Information Response Examples

```
<OFX>
   <SIGNONMSGSRSV2> <!--Signon Response->
      <SONRS>
         <STATUS>
            <CODE>0
            <SEVERITY>INFO
         </STATUS>
         <DTSERVER>19970307081437         <!--Timestamp, 3/07/97, 8:14:37am-->
         <LANGUAGE>ENG
         <DTPROFUP>19970301070000         <!--Timestamp, 3/01/97, 7:00:00am-->
         <DTACCTUP>19970301070000         <!--Timestamp, 3/01/97, 7:00:00am-->
      </SONRS>
   </SIGNONMSGSRSV2>
   <PRESDLVMSGSRSV1><!--Group Account Info Response-->
      <PRESGRPACCTINFOTRNRS>
         <TRNUID>10001
         <STATUS>
            <CODE>0
            <SEVERITY>INFO
         </STATUS>
         <ACCTINFORS>
            <DTACCTUP>19970122092431
            <PRESACCTINFO>
               <PRESACCTFROM>
                  <BILLPUB>PUBLISHER, INC.
                  <BILLER>415-552-9923
                  <ACCTID>408-555-4342-M132
                 <USERID>bygeorge         <!--User from group with new status-->
               </PRESACCTFROM>
               <SVCSTATUS>ACTIVE
            </PRESACCTINFO>
         </PRESACCTINFORS>
         <PRESACCTINFORS>
            <DTACCTUP>19970123082423
            <PRESACCTINFO>
               <PRESACCTFROM>
                  <BILLPUB>PUBLISHER, INC.
                  <BILLER>415-552-9923
                  <ACCTID>408-555-2341-U421
                 <USERID>132-42-5242      <!--User from group with new status-->
               </PRESACCTFROM>
               <SVCSTATUS>REJECTED
               <REASON>ACCOUNT NOT FOUND <!--User supplied account with biller-->
                                          <!--didn't match biller's records-->
            </PRESACCTINFO>
         </PRESACCTINFORS>
      </PRESGRPACCTINFOTRNRS>
   </PRESDLVMSGSRSV1>
```

```
</OFX>
```

Appendices

A. Status Codes

The following table provides a complete list of the status codes that can be returned by a server. For each status code, the table includes the following information:

- Number of the status code
- Meaning of the status code
- Conditions under which a server must return the status code

Code	*Meaning*	*Condition*
0	OK (INFO)	The server successfully processed the request.
1	Client is up-to-date (INFO)	Based on the client timestamp, the client has the latest information. The response does not supply any additional information.
2000	General error (ERROR)	Error other than those specified by the remaining error codes.
2001	Invalid account (ERROR)	
2002	General account error (ERROR)	Account error not specified by the remaining error codes.
2003	Account not found (ERROR)	The specified account number does not correspond to one of the user's accounts.
2004	Account closed (ERROR)	The specified account number corresponds to an account that has been closed.
2005	Account not authorized (ERROR)	The user is not authorized to perform this action on the account, or the server does not allow this type of action to be performed on the account.
2006	Source account not found (ERROR)	The specified account number does not correspond to one of the user's accounts.
2007	Source account closed (ERROR)	The specified account number corresponds to an account that has been closed.
2008	Source account not authorized (ERROR)	The user is not authorized to perform this action on the account, or the server does not allow this type of action to be performed on the account.
2009	Destination account not found (ERROR)	The specified account number does not correspond to one of the user's accounts.
2010	Destination account closed (ERROR)	The specified account number corresponds to an account that has been closed.
2011	Destination account not authorized (ERROR)	The user is not authorized to perform this action on the account, or the server does not allow this type of action to be performed on the account.
2012	Invalid amount (ERROR)	The specified amount is not valid for this action; for example, the user specified a negative payment amount.
2014	Date too soon (ERROR)	The server cannot process the requested action by the date specified by the user.
2015	Date too far in future (ERROR)	The server cannot accept requests for an action that far in the future.
2016	Already committed (ERROR)	The transaction cannot be canceled or modified because it has already been committed for processing.
2017	Already canceled (ERROR)	The transaction cannot be canceled or modified because it has already been canceled.

2018	Unknown server ID (ERROR)	The specified server ID does not exist or no longer exists.
2019	Duplicate request (ERROR)	A request with this <TRNUID> has already been received and processed.
2020	Invalid date (ERROR)	The specified datetime stamp cannot be parsed; for instance, the datetime stamp specifies 25:00 hours.
2021	Unsupported version (ERROR)	The server does not support the requested version.
2022	Invalid TAN (ERROR)	The server was unable to validate the TAN sent in the request.
2023	Unknown FITID (ERROR)	The specified FITID does not exist or no longer exists.
2024	Message set version not supported (ERROR)	The version of the message set specified by the client is not supported by this server.
2025	Branch ID missing (ERROR)	A <BRANCHID> value must be provided in the <BANKACCTFROM> aggregate for this country system, but this field is missing.
2026	Bank name doesn't match bank ID (ERROR)	The value of <BANKNAME> in the <EXTBANKACCTTO> aggregate is inconsistent with the value of <BANKID> in the <BANKACCTTO> aggregate.
10000	Stop check in process (INFO)	Stop check is already in process.
10500	Too many checks to process (ERROR)	The stop-payment request <STPCHKRQ> specifies too many checks.
10501	Invalid payee (ERROR)	Payee error not specified by the remaining error codes.
10502	Invalid payee address (ERROR)	Some portion of the payee's address is incorrect or unknown.
10503	Invalid payee account number (ERROR)	The account number <PAYACCT> of the requested payee is invalid.
10504	Insufficient funds (ERROR)	The server cannot process the request because the specified account does not have enough funds.
10505	Cannot modify element (ERROR)	The server does not allow modifications to one or more values in a modification request.
10506	Cannot modify source account (ERROR)	Reserved for future use.
10507	Cannot modify destination account (ERROR)	Reserved for future use.
10508	Invalid frequency (ERROR)	The specified frequency <FREQ> does not match one of the accepted frequencies for recurring transactions.
10509	Model already canceled (ERROR)	The server has already canceled the specified recurring model.
10510	Invalid payee ID (ERROR)	The specified payee ID does not exist or no longer exists.
10511	Invalid payee city (ERROR)	The specified city is incorrect or unknown.
10512	Invalid payee state (ERROR)	The specified state is incorrect or unknown.
10513	Invalid payee postal code (ERROR)	The specified postal code is incorrect or unknown.
10514	Bank payment already processed (ERROR)	The server has already processed the bank payment.
10515	Payee not modifiable by client (ERROR)	The server does not allow clients to change payee information.
10516	Wire beneficiary invalid	The specified wire beneficiary does not exist or no longer

	(ERROR)	exists.
10517	Invalid payee name (ERROR)	The server does not recognize the specified payee name.
10518	Unknown model ID (ERROR)	The specified model ID does not exist or no longer exists.
10519	Invalid payee list ID (ERROR)	The specified payee list ID does not exist or no longer exists.
10520	Payment type not supported (ERROR)	The value of <PMTTYPE> in the <PMTINFO> aggregate is not supported by this server.
10600	Unknown table type (ERROR)	The specified table type is not recognized or does not exist.
12250	Investment transaction download not supported (WARN)	The server does not support investment transaction download.
12251	Investment position download not supported (WARN)	The server does not support investment position download.
12252	Investment positions for specified date not available (WARN)	The server does not support investment positions for the specified date.
12253	Investment open order download not supported (WARN)	The server does not support open order download.
12254	Investment balances download not supported (WARN)	The server does not support investment balances download.
12500	One or more securities not found (ERROR)	The server could not find the requested securities.
13000	User ID & password will be sent out-of-band (INFO)	The server will send the user ID and password via postal mail, e-mail, or another means. The accompanying message will provide details.
13500	Unable to enroll user (ERROR)	The server could not enroll the user.
13501	User already enrolled (ERROR)	The server has already enrolled the user.
13502	Invalid service (ERROR)	The server does not support the service <SVC> specified in the service-activation request.
13503	Cannot change user information (ERROR)	The server does not support the <CHGUSERINFORQ> request.
15000	Must change USERPASS (INFO)	The user must change his or her <USERPASS> number as part of the next OFX request.
15500	Signon (for example, user ID or password) invalid (ERROR)	The user cannot signon because he or she entered an invalid user ID or password.
15501	Customer account already in use (ERROR)	The server allows only one connection at a time, and another user is already signed on. Please try again later.
15502	USERPASS lockout (ERROR)	The server has received too many failed signon attempts for this user. Please call the FI's technical support number.
15503	Could not change USERPASS (ERROR)	The server does not support the <PINCHRQ> request.
15504	Could not provide random data (ERROR)	The server could not generate random data as requested by the <CHALLENGERQ>.
15505	Country system not supported (ERROR)	The server does not support the country specified in the <COUNTRY> field of the <SONRQ> aggregate.

16500	HTML not allowed (ERROR)	The server does not accept HTML formatting in the request.
16501	Unknown mail To: (ERROR)	The server was unable to send mail to the specified Internet address.
16502	Invalid URL (ERROR)	The server could not parse the URL.
16503	Unable to get URL (ERROR)	The server was unable to retrieve the information at this URL (e.g., an HTTP 400 or 500 series error).

B. Change History

B.1 OFX 1.0 to 1.0.1

This section describes the revisions to Open Financial Exchange that occurred between version 1.0 and version 1.0.1. These revisions are grouped into the following categories:

- Revisions to the specification text, organized by chapter
- General revisions to the specification text
- Changes to the Document Type Definition (DTD) files

B.1.1 Specification Changes by Chapter

B.1.1.1 Chapter 1, "Overview"

Section	*Subject*	*Change Type*	*Change*
1.2.1	Header examples, SECURITY tag	Correction	Changed SECURITY:1 to SECURITY:Type1.

B.1.1.2 Chapter 2, "Structure"

B.1.1.2.1 Sections 2.2 to 2.4.4

Section	*Subject*	*Change Type*	*Change*
2.1	HTTP headers, errors	Clarification	The server must return code 400 for any problem that prevents it from processing the request file. Processing problems include failures relating to security, communication, parsing, or the Open Financial Exchange headers (for example, the client requested an unsupported language).
2.2, 2.2.1 to 2.2.7	Headers	Clarification	Defined values for the Open Financial Exchange headers.
2.3.1	SGML compliance	Correction	Changed "complaint" to "compliant."
2.3.2	SGML values	Clarification	Clarified acceptable SGML values.
2.3.3	Comments	Addition	For explanatory purposes, the examples in the specification contain comments. However, Open Financial Exchange files ***cannot*** contain comments.
2.4.3	Messages, transaction wrappers	Clarification	For requests, the transaction wrapper adds a transaction unique ID <TRNUID>. For responses, the transaction wrapper adds the same transaction unique ID <TRNUID>, plus a <STATUS> aggregate.
2.4.4	Message sets	Correction	Added missing "rq" and "rs" for <***XXX***MSGSRQV***n***> and <***XXX***MSGSRSV***n***>. For each message set of ***XXX*** and version ***n***, there are two aggregates – one for requests (<***XXX***MSGSRQV***n***>) and one for responses

			(<***XXX***MSGSRSV***n>***).

B.1.1.2.2 Sections 2.5.1 to 2.6

Section	*Subject*	*Change Type*	*Change*
2.5.1	<SONRQ>, user IDs in signon requests	Clarification	Servers must accept user IDs, with or without punctuation.
	<SONRS>, server response to signon failure	Clarification	If the server returns any signon error, it must respond to all other requests in the same <OFX> block with status code 15500. The server must return status code 15500 to all requests; it cannot simply ignore the requests.
2.5.1.1	<SONRQ>	Correction	Moved <SESSCOOKIE> after <FI> aggregate.
		Correction	Choose either 1) <USERID> and <USERPASS> or 2) <USERKEY>.
2.5.1.2	<SONRS>	Clarification	Updated element sizes.
2.5.2.1	<PINCHRQ>	Clarification	Clarified scope of the PIN change.
2.5.3	<SIGNONMSGSET>	Addition	Added description of the signon message set.
2.6	<GETMIMERS> example	Correction	Changed zzz.html to zzz.jpg.
2.7	Extensions to Open Financial Exchange	Clarification	End tags are required for all aggregates and elements introduced by an organization. For example, <ABC.SOMETHING> must be followed by </ABC.SOMETHING>.

B.1.1.3 Chapter 3, “Common Aggregates, Elements, and Data Types”

B.1.1.3.1 Sections 3.1.2 to 3.2.3

Section	*Subject*	*Change Type*	*Change*
3.1.2	User-supplied numbers	Addition	Clients will not attempt to strip dashes or other punctuation from user-supplied numbers, such as the <TAXID> in an enrollment request or the <ACCTTO> in a service-addition request. Servers must be prepared to accept these numbers with or without punctuation.
3.1.3	<BAL>	Correction	Balance <NAME> is of type ***A-32***.
		Correction	Moved <DTASOF> before <CURRENCY> aggregate.
		Correction	Changed format of DOLLAR to DDDD.cc.
3.1.4	<STATUS>	Clarification	<MESSAGE> element is of type ***A-255***.
		Addition	Added status code 15500 to the list of general errors that a server can return.
3.2.3	<TRNUID> element	Clarification	In most cases, clients originate transactions. When a client originates a <TRNUID> for a transaction, the value of the <TRNUID> is always set to a unique identifier. When the server originates a transaction, the value of the <TRNUID> must be set to zero.

B.1.1.3.2 Sections 3.3.1 to 3.3.4

Section	*Subject*	*Change Type*	*Change*
3.3.1, 3.3.1.1, 3.3.1.2, 3.3.1.3	Dates and times	Clarification	Clarified format of ***date***, ***datetime***, and ***time.***
3.3.2	Amounts, prices, and quantities	Clarification	Clarified size of types, implied decimal point if none is specified, and server error code for unsupported size or precision.
3.3.2.1	Amounts	Correction	Amounts that do not represent whole numbers (for example, 540.32), must include a decimal point or comma to indicate the start of the fractional amount. Amounts should not include any punctuation separating thousands, millions, and so forth.
3.3.4	***currSymbol*** type	Addition	Added ***currSymbol*** type***,*** a three-letter code that identifies the currency being used for a request or response. The currency codes are based on ISO-4217.
	URL	Clarification	Specified size of ***URL*** type as ***A-255***.

B.1.1.4 Chapter 5, “International Support”

Section	*Subject*	*Change Type*	*Change*
5.1	Language and encoding	Clarification	Servers must respond with the encoding, character set, and language requested by the client.
5.2	Currency aggregate,	Clarification	<CURRATE> element is of type ***rate.***

	<CURRATE> element		

B.1.1.5 Chapter 6, “Data Synchronization”

Section	*Subject*	*Change Type*	*Change*
6.4	<LOSTSYNC>	Correction	Synchronization responses can optionally include the <LOSTSYNC> element.
6.6	Client synchronization requests	Clarification	Synchronization requests can contain <TOKEN>, <TOKENONLY>, or <REFRESH>.
6.10.1	Lite synchronization, OLDFILEUID and NEWFILEUID	Clarification	Clarified role of OLDFILEUID and NEWFILEUID in file-based error recovery.
6.11	Synchronization example	Addition	Added <REJECTIFMISSING>N after <TOKEN>.
		Correction	Moved <TRNUID> before <STATUS> aggregate in synchronization response.

B.1.1.6 Chapter 7, “FI Profile”

Section	*Subject*	*Change Type*	*Change*
7.1.3	Batching and routing	Clarification	Signon message set is an exception to the batching and routing rule. See section 7.1.3.
7.1.4	Client signon in profile requests	Clarification	Clarified values for the first signon request (before the user has a valid user ID and password).
			Described profile to be returned before and after user enrollment.
7.1.5	<PROFRQ>	Clarification	Specified type of <DTPROFUP> element as ***datetime***.
7.2	<PROFRS>	Clarification	Clarified contents of <PROFRS> depending on whether the client has the most up-to-date profile.
		Correction	Moved <COUNTRY> after <POSTALCODE>.
	Element sizes	Clarification	Clarified size of elements
7.2.1	<MSGSETCORE>	Clarification	Clarified size and definition of elements.
7.2.2	<SIGNONINFO>	Clarification	Clarified size and definition of elements.
		Addition and deletion	Replaced <ALPHA> and <NUMERIC> with <CHARTYPE>.
7.2.3	Profile-response status codes	Addition	Added status code 1, “Client is up-to-date.”

B.1.1.7 Chapter 8, “Activation & Account Information”

Section	*Subject*	*Change Type*	*Change*
8.4.1	User IDs in signon requests	Clarification	Servers must accept user IDs with or without punctuation.
8.4.2	<ENROLLRQ>	Clarification	Clarified size and definition of elements.
8.4.3	<ENROLLRS>	Clarification	Clarified size and definition of elements.
8.5.1	<ACCTINFORQ>	Clarification	Clarified size and definition of elements.
		Correction	Made <DTACCTUP> required.
		Deletion	Deleted <INCIMAGES> element.
8.5.2	<ACCTINFORS>	Clarification	Clarified size and definition of elements.
8.5.3	<ACCTINFO>	Clarification	Clarified size and definition of elements.
		Deletion	Deleted <LOGO> element.
8.5.4	Account information status codes	Addition	Added status code 1, “Client is up-to-date.”
		Deletion	Deleted status code 13001, “No change since supplied <DTACCTUP>”.
8.5.5	Account information example	Deletion	Deleted <INCIMAGES> and <LOGO> elements from example.
8.6.1.1	<ACCTRQ>	Clarification	Clarified size and definition of elements.
8.6.1.2	<ACCTRS>	Addition	Added <SVCSTATUS> element.
8.6.1.3	<SVCADD>	Correction	Changed <ACCTTO> to <***XXX***ACCTTO>.
8.6.1.4	<SVCCHG>	Correction	Changed <ACCTTO> to <***XXX***ACCTTO>.
		Correction	Changed <ACCTFROM> to <***XXX***ACCTFROM>.
8.6.1.5	<SVCDEL>	Correction	Changed <ACCTTO> to <***XXX***ACCTTO>.
			Changed <ACCTFROM> to <***XXX***ACCTFROM>.
8.6.3	Service activation example	Addition	Added <SVCSTATUS> element to example.
8.7.1	<CHGUSERINFORQ>	Clarification	Clarified size and definition of elements.
8.7.2	<CHGUSERINFORS>	Clarification	Clarified size and definition of elements.
		Addition	Added <ADDR3> element.
8.8	<SIGNUPMSGSET>	Clarification	Clarified size and definition of elements.
		Correction	<WEBENROLL> is not required.
		Addition	Added <OTHERENROLL> aggregate.
		Addition	Added <CLIENTACTREQ> element.

B.1.1.8 Chapter 9, “Customer to FI Communication”

Section	Subject	Change Type	Change
9.2	E-mail messages	Correction	The message body can be HTML-encoded text or plain text.
9.2.2	<MAIL>	Clarification	Clarified size and definition of elements.
9.2.2.1	<INCIMAGES>	Addition	Described use of <INCIMAGES> in requests and responses.
9.2.2.2	<USEHTML>	Addition	Described use of <USEHTML> in requests and responses.
9.2.4	<MAILSYNCRQ> and <MAILSYNCRS>	Clarification	Clarified size and definition of elements.
	<MAILSYNCRQ>	Addition	Added <TOKENONLY>, <REFRESH>, <REJECTIFMISSING>, <INCIMAGES>, <USEHTML>, and <MAILTRNRQ>.
9.2.5	E-mail example	Correction	Revised example to show the use of e-mail synchronization
9.2.5.1	Example of synchronization request	Addition	Added <REJECTIFMISSING>N, <INCIMAGES>N, and <USEHTML>Y after <TOKEN>.
9.3.2	Multi-part MIME example	Correction	Changed SECURITY:1 to SECURITY:TYPE1.
9.4	<EMAILMSGSET>	Correction	Changed <EMAIL> to <MAILSUP>.
		Correction	Changed <GETMIME> to <GETMIMESUP>.
		Clarification	Clarified size and definition of elements.

B.1.1.9 Chapter 10, “Recurring Transactions”

Section	Subject	Change Type	Change
10.2	<RECURRINST>	Correction	Moved <NINSTS> element before <FREQ>.
10.2.1	<FREQ> element	Deletion	Deleted TRIANNUALLY value.
10.2.2	Repeating payment example	Correction	Moved <NINSTS> element before <FREQ> in the request and response.
10.5.1	Example of synchronization request	Addition	Added <REJECTIFMISSING>N after <TOKEN>.

B.1.1.10 Chapter 11, “Banking”

B.1.1.10.1 Sections 11.2.1.1 to 11.8.3

Section	*Subject*	*Change Type*	*Change*
11.2.1.1	<STPCHKSYNCRQ>	Addition	Added <TOKENONLY> and <REFRESH>.
11.3.1	<BANKACCTFROM>	Clarification	Clarified size and definition of elements.
11.3.2	<CCACCTFROM>	Clarification	Clarified size and definition of elements.
11.3.3	<BANKACCTINFO>	Clarification	Clarified size and definition of elements.
11.4.1.1	<STMTRQ>	Clarification	Clarified size and definition of elements.
11.4.1.2	<STMTRS>	Clarification	<CURDEF> is of type ***currsymbol.***
11.4.2.1	<CCSTMTRQ>	Clarification	<CURDEF> is of type ***currsymbol***.
11.4.2.3.1	<STMTTRN>	Clarification	Clarified size and definition of elements.
11.5.1.2	<STMTENDRS>	Clarification	<CURDEF> is of type ***currsymbol***.
11.5.2	<CLOSING>	Clarification	Clarified size and definition of elements.
11.5.4	<CCSTMTENDRS>	Clarification	<CURDEF> is of type ***currsymbol***.
11.6.1.1	<STPCHKRQ>	Correction	Changed type of <CHKNUMSTART> and <CHKNUMEND> to ***A-12***.
11.6.1.2	<STPCHKRS>	Clarification	<CURDEF> is of type ***currsymbol***.
11.6.1.2.1	<STPCHKNUM>	Clarification	Clarified size and definition of elements.
11.7.1.2	<INTRARS>	Addition	Added <XFERPRCSTS> aggregate.
		Clarification	Clarified size and definition of elements.
11.7.2	<INTRAMODRQ>	Correction	When modifying a transfer, the client must specify all of the tags within the <XFERINFO> aggregate that were specified when the transfer was created, not just the tags that the client wants to modify.
11.8.2.2	<INTERRS>	Addition	Added <XFERPRCSTS> aggregate.
		Clarification	Clarified size and definition of elements.
11.8.3	<INTERMODRQ>	Correction	When modifying a transfer, the client must specify all of the tags within the <XFERINFO> aggregate that were specified when the transfer was created, not just the tags that the client wants to modify.

B.1.1.10.2 Sections 11.9.1.1 to 11.10.6.2

Section	*Subject*	*Change Type*	*Change*
11.9.1.1	<WIRERQ>	Clarification	Clarified size and definition of elements.
		Correction	Made <WIREDESTBANK> aggregate optional.
11.9.1.1.1	<WIREBENEFICIARY>	Clarification	Clarified size and definition of elements.
11.9.1.1.2	<EXTBANKDEST>	Clarification	Clarified size and definition of elements.
11.9.1.2	<WIRERS>	Clarification	Clarified size and definition of elements.
11.10.2.1	<MODPENDING>	Clarification	If the client sets this flag, the server must modify pending and future transfers.
11.10.2.2	<MODPENDING>	Clarification	Y if client requested that the server modify pending and future transfers. N if the client did not request that the server modify pending and future transfers. ***Boolean***
11.10.3.1	<CANPENDING>	Clarification	If the client sets this flag, the server must cancel pending and future transfers.
11.10.3.2	<CANPENDING>	Clarification	Y if the client requested that the server cancel pending and future transfers. N if the client did not request that the server cancel pending and future transfers.
11.10.5.1	<MODPENDING>	Clarification	If the client sets this flag, the server must modify pending and future transfers.
11.10.5.2	<MODPENDING>	Clarification	Y if client requested that the server modify pending and future transfers. N if the client did not request that the server modify pending and future transfers. ***Boolean***
11.10.5.1	<RECINTERMODRQ>	Clarification	Clarified size and definition of elements.
11.10.6.1	<CANPENDING>	Clarification	If the client sets this flag, the server must cancel pending and future transfers.
11.10.6.2	<CANPENDING>	Clarification	Y if the client requested that the server cancel pending and future transfers. N if the client did not request that the server cancel pending and future transfers.

B.1.1.10.3 Sections 11.12.1.1 to 11.12.7.2

Section	*Subject*	*Change Type*	*Change*
11.12.1.1	<STPCHKSYNCRQ>	Clarification	Clarified size and definition of elements.
		Addition	Added <TOKENONLY>, <REFRESH>, and <REJECTIFMISSING>.
11.12.1.2	<STPCHKSYNCRS>	Clarification	Clarified size and definition of elements.
11.12.2.1	<INTRASYNCRQ>	Clarification	Clarified size and definition of elements.
		Addition	Added <TOKENONLY>, <REFRESH>, and <REJECTIFMISSING>.
11.12.2.2	<INTRASYNCRS>	Clarification	Clarified size and definition of elements.
11.12.3.1	<INTERSYNCRQ>	Clarification	Clarified size and definition of elements.
		Addition	Added <TOKENONLY>, <REFRESH>, and <REJECTIFMISSING>.
11.12.3.2	<INTERSYNCRS>	Clarification	Clarified size and definition of elements.
11.12.4.1	<WIRESYNCRQ>	Clarification	Clarified size and definition of elements.
		Addition	Added <TOKENONLY>, <REFRESH>, and <REJECTIFMISSING>.
11.12.4.2	<WIRESYNCRS>	Clarification	Clarified size and definition of elements.
11.12.5.1	<RECINTRASYNCRQ>	Clarification	Clarified size and definition of elements.
		Addition	Added <TOKENONLY>, <REFRESH>, and <REJECTIFMISSING>.
11.12.5.2	<RECINTRASYNCRS>	Clarification	Clarified size and definition of elements.
11.12.6.1	<RECINTERSYNCRQ>	Clarification	Clarified size and definition of elements.
		Addition	Added <TOKENONLY>, <REFRESH>, and <REJECTIFMISSING>.
11.12.6.2	<RECINTERSYNCRS>	Clarification	Clarified size and definition of elements.
11.12.7.1	<BANKMAILSYNCRQ>	Clarification	Clarified size and definition of elements.
		Addition	Added <REJECTIFMISSING>, <INCIMAGES>, and <USEHTML> elements.
		Correction	The account-from aggregate can be either <BANKACCTFROM> or <CCACCTFROM>.
11.12.7.2	<BANKMAILSYNCRS>	Clarification	Clarified size and definition of elements.
		Correction	The account-from aggregate can be either <BANKACCTFROM> or <CCACCTFROM>.

B.1.1.10.4 Sections 11.13.1 to 11.14.4

Section	*Subject*	*Change Type*	*Change*
11.13.1, 11.13.1.1, 11.13.1.2, 11.13.1.3, 11.13.1.4	Bank message sets and messages	Clarification	Updated tables to identify the message set aggregates and corresponding messages.
11.13.2	<BANKMSGSET>	Correction	Made <INVALIDACCTTYPE>, <PROCDAYSOFF>, <XFERPROF>, and <STPCHKPROF> optional.
11.13.4	<INTERXFERMSGSET>	Correction	Made <PROCDAYSOFF> optional.
11.13.5	<WIREXFERMSGSET>	Correction	Made <PROCDAYSOFF> optional.
11.14.1	Example	Correction	Changed <BALAMT>> to <BALAMT>.
11.14.2	Recurring transfer response	Correction	Changed <CURDEF>ENG to <CURDEF>USD.
11.14.3	Example	Correction	Changed <CHKSTATUS G>0 to <CHKSTATUS>0.
		Addition	Added <FEE> and <FEEMSG> elements.
11.14.4	Example of synchronization request	Addition	Added <REJECTIFMISSING>N after <TOKEN>.

B.1.1.11 Chapter 12, “Payments”

B.1.1.11.1 Sections 12.5.1 to 12.8.2.2

Section	Subject	Change Type	Change
12.5.1	<BPACCTINFO>	Clarification	Clarified values of <SVCSTATUS> element.
12.5.2	<PMTINFO>	Clarification	Clarified size and definition of elements.
12.5.2.1	<PAYEE>	Clarification	Clarified size and definition of elements.
12.5.2.2	<EXTDPMT>	Additions	Added <EXTDPMTFOR>, <EXTDPMTCHK>, and <ADJDATE>.
		Correction	<EXTDPMT> can contain one or both of the following aggregates: <EXTDPMTINV> and <EXTDPMTDSC>.
		Clarification	Clarified size and definition of elements.
12.5.2.3	<EXTDPAYEE>	Clarification	Clarified size and definition of elements.
12.5.2.4	<PMTPRCSTS>	Clarification	Clarified size and definition of elements.
		Correction	Made <PMTPRCCODE> and <DTPMTPRC> required.
12.6.1.2	<PMTRS>	Addition	Added note regarding payee synchronization.
		Clarification	Clarified size and definition of elements.
12.6.2	<PMTMODRQ>	Correction	When modifying a payment, the client must specify all of the tags within the <PMTINFO> aggregate that were specified during the payment creation, not just the tags that it wants to modify.
12.6.2.3	<PMTMODRS>	Addition	If the <PMTMODRQ> created a new payee, the server must create and store a payee response that would be available for a payee synchronization request.
12.6.4.2	<PMTINQRS>	Clarification	Clarified size and definition of elements.
12.7.1.1	<RECPMTRQ>	Clarification	Clarified size and definition of elements.
12.7.1.2	<RECPMTRS>	Clarification	Clarified size and definition of elements.
12.7.2.1	<RECPMTMODRQ>	Clarification	Clarified size and definition of elements.
12.7.2.2	<RECPMTMODRS>	Clarification	Clarified size and definition of elements.
12.7.3.1	<RECPMTCANCRQ>	Clarification	Clarified size and definition of elements.
12.7.3.2	<RECPMTCANCRS>	Clarification	Clarified size and definition of elements.
12.8.1.1	<PMTMAILRQ>	Clarification	Clarified size and definition of elements.
12.8.1.2	<PMTMAILRS>	Clarification	Clarified size and definition of elements.
12.8.2, 12.8.2.1, 12.8.2.2	Payment mail synchronization	Addition	Added description of <PMTMAILSYNCRQ> and <PMTMAILSYNCRS> aggregates.

B.1.1.11.2 Sections 12.9.1.1 to 12.12.9

Section	*Subject*	*Change Type*	*Change*
12.9.1.1	<PAYEERQ>	Clarification	Clarified size and definition of elements.
12.9.1.2	<PAYEERS>	Correction	Moved <PAYACCT> after <EXTDPAYEE> aggregate.
12.9.2.1	<PAYEEMODRQ>	Clarification	Clarified size and definition of elements.
		Correction	Made <PAYACCT> optional.
12.9.2.2	<PAYEEMODRS>	Deletion	Deleted <PAYEEID>.
		Correction	Moved <PAYACCT> before <EXTDPAYEE> aggregate.
12.9.3.1	<PAYEEDELRQ>	Clarification	Clarified size and definition of elements.
12.9.3.2	<PAYEEDELRS>	Clarification	Clarified size and definition of elements.
12.9.4.1	<PAYEESYNCRQ>	Clarification	Clarified size and definition of elements.
		Addition	Added <TOKENONLY>, <REFRESH>, and <REJECTIFMISSING>.
12.9.4.2	<PAYEESYNCRS>	Clarification	Clarified size and definition of elements.
12.10.1.1	<PMTSYNCRQ>	Clarification	Clarified size and definition of elements.
		Addition	Added <TOKENONLY>, <REFRESH>, and <REJECTIFMISSING>.
12.10.1.2	<PMTSYNCRS>	Clarification	Clarified size and definition of elements.
12.10.2.1	<RECPMTSYNCRQ>	Correction	Changed <PMTTRNRQ> to <RECPMTTRNRQ>.
		Addition	Added <TOKENONLY>, <REFRESH>, and <REJECTIFMISSING>.
		Clarification	Clarified size and definition of elements.
12.10.2.2	<RECPMTSYNCRS>	Correction	Changed <PMTTRNRQ> to <RECPMTTRNRQ>.
		Clarification	Clarified size and definition of elements.
12.11	Message sets and profile	Correction	Changed <PMTMSGSETV1> to <BILLPAYMSGSETV1>.
12.11.2	<BILLPAYMSGSET> (was <PMTMSGSET>)	Correction	Changed <PMTMSGSET> to <BILLPAYMSGSET>.
		Correction	Made <PROCDAYSOFF> optional.
		Clarification	Clarified size and definition of elements.
12.12.9	Example of synchronization request	Addition	Added <REJECTIFMISSING>N after <TOKEN>.

B.1.1.12 Chapter 13, “Investments”

B.1.1.12.1 Sections 13 to 13.8.5.6

Section	Subject	Change Type	Change
13	Client Sends/Server Responds table	Correction	Changed Cash Sub-Account Balance to Available Cash Balance.
		Correction	Changed Short Sub-Account Balance to Short Balance.
		Correction	Changed Margin Sub-Account Balance to Margin Balance.
		Deletion	Removed Other Sub-Account Balance.
13.3	Units, precision, and sign	Clarification	Clarified definition and precision of values.
13.6.1	<INVACCTFROM>	Clarification	Clarified size and definition of elements.
	<INVACCTTO>	Addition	Contains the same elements as <INVACCTFROM>.
	<INVACCTINFO>	Clarification	Clarified size and definition of elements.
		Clarification	Clarified size and definition of elements.
13.7.1.1	<INVSTMTMSGSET>	Clarification	Clarified usage of <CANEMAIL>.
13.8.1	<SECID>	Clarification	Clarified size and definition of elements.
13.8.2.1	<SECLISTTRNRQ>	Correction	Made <SECLISTRQ> aggregate required.
		Correction	Added <TAN> element.
13.8.2.2	<SECLISTRQ>	Clarification	Clarified size and definition of elements.
		Correction	For the security identification, specify either <SECID>, <TICKER>, or <FIID>.
13.8.3.1	<SECLISTTRNRS>	Correction	Moved <CLTCOOKIE> after <STATUS> aggregate.
13.8.5.1	<SECINFO>	Clarification	Clarified size and definition of elements.
		Deletion	Removed paragraph beginning “Descriptive information is not standard…” The paragraph referred to the <SOURCE> tag, which is not used.
		Correction	Changed <NAME> element to <SECNAME>.
13.8.5.2	<DEBTINFO>	Clarification	Clarified size and definition of elements.
		Correction	Made <COUPONFREQ> an enumerated type with the following values: MONTHLY, QUARTERLY, SEMIANNUAL, ANNUAL, or OTHER.
13.8.5.3	<MFINFO>	Clarification	Clarified size and definition of elements.
13.8.5.4	<OPTINFO>	Clarification	Clarified size and definition of elements.
13.8.5.5	<OTHERINFO>	Clarification	Clarified size and definition of elements.
13.8.5.6	<STOCKINFO>	Clarification	Clarified size and definition of elements.

B.1.1.12.2 Sections 13.9.1.2 to 13.9.2.5.2

Section	Subject	Change Type	Change
13.9.1.1	<INVSTMTTRNRQ>	Addition	Added <TAN> element.
		Correction	Made <INVSTMTRQ> aggregate required.
13.9.1.2	<INVSTMTRQ>	Addition	If <DTASOF> is not included with the <INCPOS> aggregate, the server should return the most current position information available.
		Correction	Made <INCOO> element and <INCPOS> aggregate required.
		Clarification	Clarified size and definition of elements.
13.9.2.1	<INVSTMTTRNRS>	Correction	Moved <CLTCOOKIE> after <STATUS> aggregate.
13.9.2.2	<INVSTMTRS>	Clarification	Clarified size and definition of elements.
13.9.2.3	<INVBANKTRAN>	Correction	All elements are required.
13.9.2.4.1	<INVTRAN>	Clarification	Clarified size and definition of elements.
13.9.2.4.2	Transaction aggregate elements	Addition	Added <DTPURCHASE>.
		Correction	Changed <AVECOSTBASIS> to <AVGCOSTBASIS>
		Clarification	Clarified size and definition of elements.
13.9.2.4.4	Investment transaction aggregates	Clarification	<CLOSUREOPT> aggregate: Defined RELFITID.
		Correction	<SELLOPT> aggregate: Corrected order of aggregates and elements.
		Addition	<TRANSFER> aggregate: Added <AVECOSTBASIS>, <UNITPRICE>, and <DTPURCHASEDATE>.
13.9.2.4.5	Valid transactions by security type	Addition	Since JRNLFUND and MARGININTEREST do not refer to securities, there are no checks in any of the security columns for these transactions.
		Correction	Moved <PAYACCT> before <EXTDPAYEE> aggregate.
13.9.2.5.1	<OO>	Clarification	Clarified size and definition of elements.
13.9.2.5.2	Investment Open Order Aggregates	Clarification	Clarified size and definition of elements.

B.1.1.12.3 Sections 13.9.2.6.1 to 13.11

Section	Subject	Change Type	Change
13.9.2.6.1	<INVPOS>	Clarification	Clarified size and definition of elements.
13.9.2.6.2	Investment positions	Clarification	Clarified size and definition of elements.
13.9.2.7	<INVBAL>	Clarification	Clarified aggregate description.
			Clarified size and definition of elements in table.
		Correction	Changed <CASHACCT> to <AVAILCASH>.
			Changed <MARGINACCT> to <MARGINBALANCE>.
			Changed <SHORTACCT> to <SHORTBALANCE>.
		Deletion	Deleted <OTHERACCT>.
13.10.2, 13.10.2.1, 13.10.2.2	Investment e-mail synchronization	Addition	Added description of requests and responses.
13.10.2.1	<INVMAILSYNCRQ>	Addition	Added <TOKENONLY>, <REFRESH>, and <REJECTIFMISSING>.
13.11	Example	Correction	Changed <BALTYPE>P to <BALTYPE>PERCENT.
		Deletion	Deleted <DTEND> element, since the example is requesting transactions up to the current date.
		Correction	Changed <CASHACCT> to <AVAILCASH>.
		Correction	Changed <MARGINACCT> to <MARGINBALANCE>.
		Correction	Changed <SHORTACCT> to <SHORTBALANCE>.
		Deletion	Deleted <OTHERACCT>.
		Correction	Changed <NAME> to <SECNAME> in all <SECINFO> aggregates.

B.1.1.13 Appendix A

Appendix A, “Status Codes,” has been added to the specification. It provides a complete list of the status codes that a server can return.

B.1.2 General Specification Changes

The text associated with status codes has been standardized throughout the specification. The text for a status code should always be the same, regardless of the context in which it is used. For a complete list of the status codes that can be returned by a server, refer to Appendix A, “Status Codes.”

B.1.3 DTD Changes

The corrections to the DTD files ensure that the DTD files match the specification. They do not represent changes to the specification itself.

Subject	Change Type	Change
<ACCTINFO>	Correction	Made <DESC> and <PHONE> optional.
<ACCTRS>	Addition	Added <SVCSTATUS>.
<BANKMSGSETV1>	Correction	Made <XFERPROF> and <STOPCHKPROF> optional.

<BANKMSGSRQV1>, <BANKMSGSRSV1>	Addition	Some of the synchronization requests and responses were not listed as being part of <BANKMSGSRQV1> and <BANKMSGSRSV1>.
<CLTCOOKIE>, <TAN>, <PINCH>, <REINVDIV>	Correction	Made elements optional.
<DEBTINFO>	Correction	Made <COUPONFREQ> an enumerated type with the following values: MONTHLY, QUARTERLY, SEMIANNUAL, ANNUAL, or OTHER.
<EXTDPMT>	Correction	<EXTDPMT> can contain one or both of the following aggregates: <EXTDPMTDSC> and <EXTDPMTINV>.
	Correction	Made <DISCOUNT>, <DSCDATE>, <ADJUSTMENT>, and <ADJNO> optional.
	Correction	Added <DSCDESC> and <ADJDATE>.
	Addition	Added <EXTDPMTFOR> and <EXTDPMTCHK>.
Investment statements	Correction	Removed required ordering constraint for transactions, open orders, positions, and security lists.
<INVSTMTMSGSET>	Addition	Added <CANEMAIL> element to DTD.
<LOSTSYNC>	Addition	Added <LOSTSYNC> to DTD.
<MAILSYNCRQ>	Addition	Added <INCIMAGES> and <USEHTML>.
<MAILSYNCRS>	Deletion	Removed <INCIMAGES> and <USEHTML>.
<PAYEEMODRQ>	Correction	If <BANKACCTTO> is used, <PAYEE> is required.
<PAYEEMODRS>	Correction	Made <BANKACCTTO> and <PAYEE> optional.
<PROFRS>	Addition	Added <COUNTRY> after <POSTALCODE>.
<SECINFO>	Correction	Changed <NAME> element to <SECNAME>.
<SECRQ>	Correction	Corrected aggregate to match specification.
<SIGNONINFO>	Correction	Replaced <ALPHA> and <NUMERIC> with <CHARTYPE>.
<SIGNUPMSGSRQV1>, <SIGNUPMSGSRSV1>,	Addition	Added <CHGUSERINFOSYNCRQ> and <CHGUSERINFOSYNCRS>.
<SVCADD>	Deletion	Deleted <SVCSTATUS>.
<SVCCHG>	Deletion	Deleted <SVCSTATUS>.

Subject	*Change Type*	*Change*
<SVCCHG>	Deletion	Deleted <SVCSTATUS>.
<TAXEXEMPT>	Correction	Corrected spelling of tag.
Unused elements	Deletion	Removed elements that were not used in any aggregates. ERROR, DESCRIPTION, AMOUNT, PAYEEACCTTO, GLOBALBUDGET, and LOGO.
<***XXX***SYNCRQ>	Addition	Added <TOKENONLY>, <REJECTIFMISSING>, <REFRESH>.
<***XXX***SYNCRS>	Addition	Added <LOSTSYNC> element.
<***XXX***TRNRS>	Correction	Made <***XXX***RS> optional.

B.2 OFX 1.0.1 to 1.0.2

This section describes the revisions to Open Financial Exchange that occurred between version 1.0.1 and version 1.0.2. These revisions are grouped into the following categories:

- Revisions to the specification text, organized by chapter
- General revisions to the specification text
- Changes to the Document Type Definition (DTD) files

B.2.1 Specification Changes by Chapter

B.2.1.1 Chapter 1, "Overview"

Section	*Subject*	*Change Type*	*Change*
1.2	OFX headers, VERSION	Update	Updated VERSION:101 to VERSION:102 to indicate version 1.0.2 of the DTDs.

B.2.1.2 Chapter 2, "Structure"

B.2.1.2.1 Sections 2 to 2.4.5

Section	*Subject*	*Change Type*	*Change*
2.	Data blocks	Correction	Open Financial Exchange data consists of some headers plus only ***one*** Open Financial Exchange data block.
2.1	HTTP headers	Addition	If a communication time-out error occurs while an OFX server and back-end server are communicating to fill a request, then the server MUST return code 500.
2.2, 2.2.3	OFX headers, VERSION	Update	Updated VERSION:101 to VERSION:102 to indicate version 1.0.2 of the DTDs.
2.2.4	Security	Deletion	Reference to Type 2 security has been deleted.
2.4.2	Case sensitivity	Addition	Upper case letters must be used for element names and enumerated values.
2.4.3	Top level	Clarification	A single file must contain only one OFX block.
2.4.5.3	Message Set Version Numbers	Addition	Added section about message set version numbers. This section distinguishes message set version numbers from the version numbers of the OFX headers and the DTD files.
2.4.6	Message sets and version control	Addition	List of status code values for the <CODE> element of <STATUS>: 0 Success (INFO) 2000 General error (ERROR) 2022 Invalid TAN (ERROR)

B.2.1.2.2 Sections 2.5.1.1 to 2.5.2.3

Section	Subject	Change Type	Change
2.5.1.1	<SONRQ>	Clarification	The effective size of <USERPASS> is A-32. However, if Type 1 security is used, then the actual length is A-171.
2.5.1.1	<FI>	Clarification	The client will determine out-of-band whether a FI aggregate should be used and if so, the appropriate values for it. If the FI aggregate is to be used, then the client should send it in every request, and the server should return it in every response.
2.5.1.2	<SONRS>	Clarification	A client should use <DTPROFUP> and <DTACCTUP> only when the service provider that originated <SONRS> is the same provider that is specified by <SPNAME> in the profile message set. A client can determine if the service provider is the same by comparing the value of <SPNAME> in the appropriate message set with the value for <SPNAME> in the profile message set.
2.5.2.1	<PINCHRQ>	Clarification	A <USERPASS> change request changes the customer's password for the specific realm associated with the messages contained in the OFX block.
2.5.2.1	<NEWUSERPASS>	Clarification	The effective size of <NEWUSERPASS> is A-32. However, if Type 1 security is used, then the actual field length is A-171.
2.5.2.3	<CHALLENGERQ> <CHALLENGERS>	Addition	Information on the challenge request and challenge response, the first steps in Type 1 security, has been added.
2.5.2.3	<CHALLENGERQ> <CHALLENGERS>	Addition	Status code 15501, Could not provide random data (ERROR), has been added to the list of Status code values for the <CODE> element of <STATUS>.
2.5.3	<SIGNONMSGSET>	Deletion	Deleted <PINCH> and <CHGPINFIRST> from <SIGNONMSGSET>.

B.2.1.3 Chapter 3, "Common Aggregates, Elements, & Data Types"

Section	Subject	Change Type	Change
3.2.8.2	Date and datetime	Addition	Time zones are specified by an offset and optionally, a time zone name. The offset defines the time zone.

B.2.1.4 Chapter 4, "Open Financial Exchange Security"

Section	Subject	Change Type	Change
Entire chapter	Security	Addition	The previous Chapter 4 on Open Financial Exchange Security has been replaced by a new one.

B.2.1.5 Chapter 7, "FI Profile"

Section	Subject	Change Type	Change
7.2	Profile response	Deletion	<SYNCMODE> AND <RESPFILEER> were deleted from <PROFRS> and added to <MSGSETCORE>.
7.2.1	Profile response	Clarification	<SPNAME> was added to <MSGSETCORE> with a

			note indicating that if a FI out-sources its OFX server to a service provider, then the SPNAME element should be included in the <MSGSETCORE>.
7.2.1	Message set	Addition	<SYNCMODE> AND <RESPFILEER> were added to <MSGSETCORE> and deleted from <PROFRS>.
7.2.1	<TRANSPSEC>	Clarification	For all message sets currently defined in Open Financial Exchange, the value of <TRANSPSEC> must be YES.
7.2.1	<TRANSPSEC>	Deletion	Type 2 has been deleted as an enumerated type for <TRANSPSEC>.
7.2.1	<VER>	Correction	Version number of the message set, (for example, <VER>1 for version 1 of the message set), ***N-5***
7.2.2	Signon realms	Addition	<PINCH> and <CHGPINFIRST> have been added to <SIGNONINFO>.

B.2.1.6 Chapter 8, "Activation & Account Information"

Section	Subject	Change Type	Change
8.5	Account information	Clarification	Requests and responses include a <DTACCTUP> element. Responses contain the last time a server updated the information. Clients are REQUIRED to send this in a subsequent request, and servers are REQUIRED to compare this to the current modification time and only send information if it is more recent.
8.6.1.1	Request <ACCTRQ>	Clarification	The components of <ACTION> are described in more detail.
8.6.1.2	Response <ACCTRS>	Clarification	The components of <ACTION> are described in more detail.
8.7	<CHGUSERINFOTRNRQ> and <CHGUSERINFOTRNRS>	Correction	In the first sentence of the second paragraph, the incorrect tag, <CHGUSERINFOTRNRSRQ>, was changed to <CHGUSERINFOTRNRS>.

B.2.1.7 Chapter 9, "Customer to FI Communication"

Section	Subject	Change Type	Change
9.3.2	OFX headers, VERSION	Update	Updated VERSION:101 to VERSION:102 to indicate version 1.0.2 of the DTDs.

B.2.1.8 Chapter 11, "Banking"

Section	Subject	Change Type	Change
11.4.2.3.1	<STMTTRN>	Correction	<FITID> is of type ***FITID***.
11.5.4.2	<CCCLOSING>	Correction	<FITID> is of type ***FITID***.
11.6.1.1	Stop check	Change	<NAME> type changed from A-255 to A-32.
11.6.1.1.2	Stop check	Change	<NAME> type changed from A-255 to A-32.
11.6.1.1.2.1	Stop check	Change	<NAME> type changed from A-255 to A-32.
11.12.5.1	Statement closing download	Change	<RECINTRATRANRQ>has been changed to optional.

B.2.1.9 Chapter 12, "Payments"

Section	Subject	Change Type	Change

12.5.2.3	<EXTDPAYEE>	Change	<PAYEEID> type changed from N-12 to A-12.
12.5.2	<PMTINFO>	Clarification	More detailed information on <PAYEEID> has been added.
12.5.2	<PMTINFO>	Correction	<PAYEELSTID> is of type A-12.
12.5.2	<PMTINFO>	Clarification	<**DTDUE**> is the payment due date or the date by which payment must be received by payee, ***datetime***
12.6.1.2	<PMTRS>	Change	<PAYEELSTID> type changed from N-12 to A-12.
12.7.1.2	<RECPMTRS>	Change	<PAYEELSTID> type changed from N-12 to A-12.
12.9.1.1	<PAYEERQ>	Change	<PAYEEID> type changed from N-12 to A-12.
12.11.2	<BILLPAYMSGSET>	Clarification	<DAYSWITH> is an offset to withdrawal date, such that (DTDUE - DAYSTOPAY) + (DAYSWITH) determines the date on which the funds are withdrawn from the user's account. NOTE: If the value of <DAYSWITH> is "-1," then the withdrawal date is the same as the payment date (DTDUE).

B.2.1.10 Chapter 13, "Investments"

Section	Subject	Change Type	Change
13.8.3.3	<SECLISTRS>	Clarification	The security list response aggregate, <SECLISTRS>, is the only empty aggregate in OFX.
13.9.2.4.2	Transaction Aggregate Elements	Correction	Deleted redundant <DEBTACTION> tag.
13.9.2.4.3	<INVBUY>/<INVSELL>	Change	<COUPONFREQ> has been changed from N-12 to enumerated type.
13.9.2.4.3	<INVBUY>/<INVSELL>	Change	<AVGCOSTBASIS>, <UNITPRICE>, and <DTPURCHASE> have been changed to optional.
13.9.2.4.4	<TRANSFER>	Clarification	Clarified <DTPURCHASE>tag name in the <TRANSFER> aggregate.

B.2.1.11 Appendix A

Subject	Change Type	Change
Status code, 2021	Addition	The server does not support the requested version.
Status code, 2022	Addition	The server was unable to validate the TAN sent in the request.
Status code, 15504	Addition	Could not provide random data

B.2.2 General Specification Changes

The text item "PIN" has been replaced with "USERPASS." This change has not been made to tags containing "PIN."

B.2.3 DTD Changes

The corrections to the DTD files ensure that the DTD files match the specification. They do not necessarily represent changes to the specification itself.

File	Subject	Change Type	Change
OFXBANK.DTD	<RECINTRASYNCRQ>	Correction	Made RECINTRATRNRQ zero or more.
OFXBANK.DTD	<CHKNUMSTART>	Correction	Changed to IDTYPE.

OFXBANK.DTD	<CHKNUMEND>	Correction	Changed to IDTYPE.
OFXBILL.DTD	<CHECKNUM>	Correction	Changed to IDTYPE.
OFXINV.DTD	<COUPONFREQ>	Correction	Changed to STRTYPE.
OFXINV.DTD	<TRANSFER>	Correction	Changed AVGCOSTBASIS, UNITPRICE, DTPURCHASE to optional.
OFXPROF.DTD	<PROFRS>	Move	Moved SYNCMODE and RESPFILEER from PROFRS to MSGSETCORE.
OFXPROF.DTD	<MSGSETCORE>	Move	Moved SYNCMODE and RESPFILEER from PROFRS to MSGSETCORE.
OFXPROF.DTD	<MSGSETCORE>	Addition	Added SPNAME.
OFXPROF.DTD	<SIGNONINFO>	Move	Moved PINCH and CHGPINFIRST from SIGNONMSGSETV1 to SIGNONINFO.
OFXSIGN.DTD	<SIGNONMSGSETV1>	Move	Moved PINCH and CHGPINFIRST from SIGNONMSGSETV1 to SIGNONINFO.
OFXSIGN.DTD	<SIGNONMSGSRQV1>	Addition	Added CHALLENGETRNRQ.
OFXSIGN.DTD	<SIGNONMSGSRSV1>	Addition	Added CHALLENGETRNRS.
OFXSIGN.DTD	<CHALLENGETRNRQ>	Addition	New request.
OFXSIGN.DTD	<CHALLENGETRNRS>	Addition	New response.
OFXSIGN.DTD	<SONRS>	Change	Made DTPROFUP and DTACCTUP optional to accommodate service providers.
OFXSIGN.DTD	<CHALLENGERQ>	Addition	New request.
OFXSIGN.DTD	<CHALLENGERS>	Addition	New response.
OFXSIGN.DTD	<NONCE>	Addition	Added new element.
OFXSIGN.DTD	<FICERTID>	Addition	Added new element.

C. Errors and Omissions

This document describes the errors and omissions in the Open Financial Exchange version 1.5 specification. The clarifications are collected in lieu of republishing the specification. There are no changes to the DTD because of these corrections. OFX implementers should refer to the DTD in cases of a discrepancy between the spec and the DTD, in other words “the DTD rules”.

Specification Clarifications by Section

Section	Subject	Change Type	Change

D. Tag Index

Page numbers in **bold** indicate a section of the specification devoted to defining the element or aggregate. Page numbers in *italics* indicate an example using the element or aggregate.

<ABC.SOMETHING>.....24, B-3
<ACCOUNT>.....278
<ACCRDINT>.....255, 258, 259
<ACCTBAL>.....286, 287
<ACCTEDITMASK>..275, 276, *299*, *300*, *301*
<ACCTFORMAT>.....275, 276, *299*, *300*, *301*
<ACCTFROM>.....B-7
<ACCTID>*4*, *56*, *57*, *74*, *77*, *92*, *93*, *94*, *95*, 98, 99, 101, 106, *165*, *166*, *167*, *168*, *169*, *170*, *171*, *172*, *173*, *174*, *221*, *222*, *223*, *224*, *225*, *226*, *227*, *229*, *230*, *231*, *234*, *235*, *236*, 240, *269*, 279, *303*, *305*, *306*, *307*, *310*
<ACCTINFO>.72, 73, *74*, 241, 281, B-7, B-18
<ACCTINFORQ>**72**, *73*, 278, 281, 282, *309*, B-7
<ACCTINFORS>**72**, *74*, 240, 278, 281, 282, *310*, B-7
<ACCTINFOTRNRQ>.....72, *73*, 278
<ACCTINFOTRNRS>.....72, *73*
<ACCTKEY>.....98, 99, 101, 106
<ACCTREQUIRED>.....79, 80
<ACCTRQ>....**74**, *77*, 80, 280, *303*, B-7, B-22
<ACCTRS>.**75**, *77*, 281, *303*, B-7, B-18, B-22
<ACCTSYNCRQ>.....77
<ACCTSYNCRS>.....77
<ACCTSYNRQ>.....281
<ACCTTO>.....B-4, B-7
<ACCTTRNRQ>.....74, *77*, *303*
<ACCTTRNRS>.....75, *77*, *303*
<ACCTTYPE>*4*, *56*, *57*, *74*, *77*, *92*, *93*, *94*, *95*, 98, 99, **100**, *165*, *166*, *167*, *168*, *169*, *170*, *171*, *172*, *173*, *174*, *221*, *222*, *223*, *224*, *225*, *226*, *227*, *229*, *230*, *231*, *234*, *235*, *236*
<ACCTTYPE2>.....99, **100**
<ACTION>.....B-22
<ACTIVITY>.....286, 287
<ADDR1>62, 69, *71*, 78, 79, 129, 186, *222*, *225*, *226*, *227*, *233*, *235*, 274, 275, 279, *299*, *300*, *301*, *302*, *303*
<ADDR2>62, 69, 78, 79, 129, 186, *233*, 275, 276, 279, *300*
<ADDR3>62, 69, 78, 79, 129, 186, 275, 276, 279, B-7
<ADJAMT>.....187, 188
<ADJDATE>.....188, B-13, B-18
<ADJDESC>.....188
<ADJNO>.....188, B-18
<ADJUSTMENT>.....188, B-18
<AGGREGATE>.....5
<ALPHA>.....B-6, B-18
<ALPHAVALUE>.....5
<AMTDUE>.....286, *305*, *306*, *307*
<APPID>*4*, 20, *165*, *167*, *168*, *170*, *172*, *221*, *223*, *224*, *226*, *227*, *228*, *229*, *230*, *232*, *233*, *234*, *268*, *301*, *303*, *304*, *306*, *309*
<APPVER>*4*, 20, *165*, *167*, *168*, *170*, *172*, *221*, *223*, *224*, *226*, *227*, *228*, *229*, *230*, *232*, *233*, *234*, *268*, *301*, *303*, *304*, *306*, *309*
<ASSETCLASS>.....249, 250, 251, *271*, *272*
<AUCTION>.....263
<AVAILACCTS>.....80, 278
<AVAILBAL>.....45, 106, 107, *166*
<AVAILCASH>.....265, *270*, B-17
<AVECOSTBASIS>.....B-16
<AVGCOSTBASIS>.....255, 259, B-16, B-23
<BAL>.....**25**, 45, 254, 265, *270*, B-4
<BALAMT>.....106, 107, *166*, B-12
<BALCLOSE>.....111, 113
<BALDNLD>.....242, 243
<BALLIST>.....254, 265, *270*
<BALMIN>.....111
<BALOPEN>.....111, 113
<BALTYPE>.....25, *271*, B-17
<BANKACCTFROM>*4*, 45, *56*, *57*, 70, *74*, 76, *92*, *93*, *94*, *95*, **97**, 101, 102, 105, 110, 111, 115, 116, 119, 124, 128, 130, 134, 139, 142, 143, 144, 145, 146, 147, 148, 149, 150, 151, 152, 153, 154, 161, 162, *165*, *166*, *167*, *168*, *169*, *170*, *171*, *172*, *173*, *174*, 182, 183, 184, 185, 193, 200, 213, 214, *221*, *222*, *223*, *224*, *225*, *226*, *227*, *229*, *230*, *231*, *234*, *235*, *236*, A-2, B-9, B-11
<BANKACCTINFO>.....73, *74*, **101**, B-9
<BANKACCTTO>76, *77*, **97**, 98, 99, 100, 102, 109, 129, *167*, *168*, *170*, *171*, *174*, 179, 183, 185, 207, 209, A-2, B-18
<BANKBRANCH>.....100
<BANKCITY>.....100
<BANKID>*4*, *56*, *57*, *74*, *77*, *92*, *93*, *94*, *95*, 97, 98, 129, *165*, *166*, *167*, *168*, *169*, *170*, *171*, *172*, *173*, *174*, *221*, *222*, *223*, *224*, *225*, *226*, *227*, *229*, *230*, *231*, *234*, *235*, *236*, A-2
<BANKMAILRQ>.....**142**
<BANKMAILRS>.....**143**
<BANKMAILSYNCRQ>.....**152**, B-11
<BANKMAILSYNCRS>.....**153**, B-11
<BANKMAILTRNRQ>.....142, 153
<BANKMAILTRNRS>.....143, 144, 154
<BANKMSGSET>.....155, 156, **161**, B-12
<BANKMSGSETV1>15, 60, 155, 156, **161**, B-18
<BANKMSGSETV2>.....15, 155, 157, **161**
<BANKMSGSRQV1>*4*, *14*, 155, *165*, *167*, *168*, *170*, *172*, B-18

<BANKMSGSRQV2>.....155
<BANKMSGSRSV1>156, *166*, *167*, *169*, *171*, *173*, B-18
<BANKMSGSRSV2>.....157
<BANKNAME>.....100, A-2
<BANKPOSTALCODE>.....100
<BANKTRANLIST>.....105, 107, *166*
<BILLDETAILROW>289, **290**, 292, *305*, *306*, *307*, *308*
<BILLDETAILTABLE>289, 290, 291, 292, *305*, *307*, *308*
<BILLDETAILTABLENAME>.....289
<BILLDETAILTABLETYPE>289, 290, 291, 292, *305*, *307*, *308*
<BILLER>.....*303*, *310*
<BILLERID>274, 275, 279, *299*, *300*, *301*, *305*, *306*, *307*
<BILLERINFO>..275, 283, 284, *299*, *300*, *301*
<BILLERINFOURL2>.....276
<BILLID>285, 286, 289, 292, 293, *305*, *306*, *307*
<BILLPAYMSGSET>23, 216, 217, **219**, B-14, B-23
<BILLPAYMSGSETV1>15, 60, 216, 217, 219, B-14
<BILLPAYMSGSETV2>.....15, 216, 218, 220
<BILLPAYMSGSRQV1>216, *221*, *223*, *224*, *226*, *227*, *228*, *229*, *230*, *232*, *233*, *234*
<BILLPAYMSGSRQV2>.....217
<BILLPAYMSGSRSV1>217, *222*, *224*, *225*, *226*, *228*, *230*, *231*, *232*, *233*, *235*
<BILLPAYMSGSRSV2>.....218
<BILLPUB>275, 279, 285, *299*, *300*, *301*, *303*, *304*, *305*, *306*, *307*, *309*, *310*
<BILLREFINFO>183, 185, 286, 294, *305*, *306*, *307*
<BILLTBLSTRUCTRQ>.....292
<BILLTBLSTRUCTRS>.....**292**
<BILLTBLSTRUCTTRNRQ>.....292
<BILLTBLSTRUCTTRNRS>.....292
<BODY>.....*87*, *88*
<BOOKINGTEXT>.....185
<BPACCTINFO>.....**182**, B-13
<BRANCHID>.....98, 99, A-2
<BRAND>.....283, *299*, *300*, *301*
<BROKERID>.....240, *269*
<BUYDEBT>.....258
<BUYMF>.....258
<BUYOPT>.....258
<BUYOTHER>.....240, 258
<BUYPOWER>.....265
<BUYSTOCK>.....258, *269*
<BUYTYPE>.....255, 258, 263, *270*, *271*
<C>.....290, 292, *305*, *306*, *307*, *308*
<CALLPRICE>.....249
<CALLTYPE>.....249
<CANADDPAYEE>.....219, 220
<CANBILLPAY>.....164
<CANCELWND>.....164
<CANEMAIL> 163, 242, 243, 299, B-15, B-18
<CANMODMDLS>.....162, 219, 220
<CANMODPMTS>.....219, 220
<CANMODXFERS>.....162
<CANMULTI>.....164
<CANNOTIFY>.....163, 299
<CANPENDING>*94*, *95*, 136, 141, 172, *173*, 202, 203, *232*, B-10
<CANRECUR>.....162
<CANSCHED>.....162, 164, 165
<CANSUPPORTIMAGES>.....298
<CANUSEDESC>.....163
<CANUSERANGE>.....163
<CASESEN>.....65
<CASHACCT>.....B-17
<CCACCTFROM>97, **101**, 102, 106, 107, 112, 113, 142, 143, 146, 147, 148, 150, 151, 152, 153, B-9, B-11
<CCACCTINFO>.....**101**
<CCACCTTO>.....101, 102, 109
<CCCLOSING>.....113, B-22
<CCSTMTENDRQ>.....**112**
<CCSTMTENDRS>.....**112**, 113, B-9
<CCSTMTENDTRNRQ>.....112
<CCSTMTENDTRNRS>.....112
<CCSTMTRQ>.....106, 113, 114, B-9
<CCSTMTRS>.....**107**
<CCSTMTTRNRQ>.....106
<CCSTMTTRNRS>.....107
<CHALLENGERQ>**22**, 41, 42, A-4, B-21, B-24
<CHALLENGERS>..... **22**, 41, 42, B-21, B-24
<CHALLENGETRNRQ>.....22, B-24
<CHALLENGETRNRS>.....22, B-24
<CHARTYPE>.....65, B-6, B-18
<CHE.PTTACCTID>.....100
<CHECKING>.....241
<CHECKNUM>108, 115, 116, 144, *166*, *169*, 191, 196, *223*, *229*, B-24
<CHGPINFIRST>.....65, B-21, B-22
<CHGUSERINFO>.....80
<CHGUSERINFORQ>.....78, A-3, B-7
<CHGUSERINFORS>..... **78**, B-7
<CHGUSERINFOSYNCRQ>.....78, B-18
<CHGUSERINFOSYNCRS>.....78, B-18
<CHGUSERINFOTRNRQ>.....78, B-22
<CHGUSERINFOTRNRS>.....78, B-22
<CHGUSERINFOTRNRSRQ>.....B-22
<CHKANDDEB>.....112
<CHKDESC>.....115
<CHKERROR>.....116
<CHKMAILRS>.....**144**
<CHKNUMEND>.....115, *168*, B-9, B-24
<CHKNUMSTART>.....115, *168*, B-9, B-24
<CHKRANGE>.....115, *168*
<CHKSTATUS G>.....B-12
<CHKSTATUS>.....116, *169*, B-12

<CITY>62, 69, *71*, 78, 79, 129, 186, *222*, *225*, *226*, *227*, *233*, *235*, 275, 276, 279, *299*, *300*, *301*, *302*, *303*
<CLIENTACTREQ>.....80, B-7
<CLIENTENROLL>.....79, 80
<CLIENTROUTING>.....61
<CLOSING>.....111, B-9
<CLOSINGAVAIL>.....161, 162, 163
<CLOSUREOPT>.....258, B-16
<CLTCOOKIE>16, 126, 127, 246, 247, 252, 253, 282, 288, B-15, B-16, B-18
<CODE>*13*, 17, 21, 22, *23*, 26, 49, 52, *71*, *74*, *77*, *87*, *88*, *89*, *95*, *166*, *167*, *168*, *169*, *170*, *171*, *173*, *174*, *175*, *222*, *223*, *224*, *225*, *226*, *227*, *228*, *229*, *230*, *231*, *232*, *233*, *234*, *235*, *269*, *302*, *303*, *304*, *307*, *310*, B-20, B-21
<COLDEF>.....292
<COLNAME>.....292
<COLTYPE>.....292
<COMMISSION>.....255, 256, 257, 259, *270*
<CONFMSG>.....130
<CONSUPOSTALCODE>.....274, 275
<CORRECTACTION>.....108
<CORRECTFITID>.....108
<COUNTRY>19, 20, 62, 69, *71*, 78, 79, 129, 185, 186, 187, 221, 275, 276, 279, *299*, *300*, *301*, *302*, *303*, A-4, B-6, B-18
<COUNTSYS>19, 20, 185, 186, 187, 221, A-4
<COUPONFREQ>249, B-15, B-18, B-23, B-24
<COUPONRT>.....249
<COUTNRY>.....*304*
<CREDITCARDMSGSET>.....157, 158, 163
<CREDITCARDMSGSETV1>15, 157, 158, 163
<CREDITCARDMSGSETV2>15, 157, 158, 163
<CREDITCARDMSGSRQV1>.....157
<CREDITCARDMSGSRQV2>.....158
<CREDITCARDMSGSRSV1>.....158
<CREDITCARDMSGSRSV2>.....158
<CREDITLIMIT>.....113
<CSPHONE>.....62
<CURDEF>**45**, 105, 107, 111, 113, 116, 118, 123, 130, *166*, *168*, *169*, *171*, *173*, *174*, 191, 198, *222*, *224*, *230*, *235*, *236*, 254, *269*, B-9, B-12
<CURRATE>.....45, B-5
<CURRENCY>25, **45**, 46, 109, 112, 114, 116, 249, 257, 258, 259, 263, 264, B-4
<CURSYM>.....45, 46
<DATEBIRTH>.....70, *71*, *302*
<DAYPHONE>.....69, *71*, 78, 79, 279, *302*
<DAYSTOPAY>185, 189, 192, 219, 220, *223*, *224*, *234*, *235*
<DAYSWITH>.....162, 219, 220, B-23
<DEBADJ>.....113
<DEBTACTION>.....B-23
<DEBTCLASS>.....249
<DEBTINFO>.....248, **249**, B-15, B-18
<DEBTTYPE>.....249
<DENOMINATOR>.....256, 259
<DEPANDCREDIT>.....111
<DEPMAILRS>.....**144**
<DESC>.....25, 73, *74*, *270*, B-18
<DETAILAVAILABLE>.....286, *306*
<DFLTDAYSTOPAY>.....162, 219, 220
<DIFFFIRSTPMT>.....219, 220
<DIFFLASTPMT>.....220, 221
<DISCOUNT>.....187, B-18
<DISCOUNT2>.....188
<DOMXFERFEE>.....164, 165
<DSCAMT>.....187, 188
<DSCDATE>.....188, B-18
<DSCDESC>.....188, B-18
<DSCRATE>.....187, 188
<DTACCTUP>20, 72, *73*, *74*, *166*, *167*, *169*, *171*, *173*, *222*, *223*, *225*, *226*, *228*, *230*, *231*, *232*, *233*, *234*, *269*, 281, 282, *302*, *303*, *309*, *310*, B-7, B-21, B-22
<DTASOF>25, 106, 107, *166*, 249, 253, 254, *269*, *271*, B-4, B-16
<DTAUCTION>.....263
<DTAVAIL>.....102, 108, 185, 221
<DTBILL>.....286, *305*, *306*, *307*
<DTCALL>.....249
<DTCHANGED>.....22
<DTCLIENT>*4*, 19, *165*, *167*, *168*, *169*, *172*, *221*, *223*, *224*, *226*, *227*, *228*, *229*, *230*, *232*, *233*, *234*, *268*, *301*, *302*, *304*, *306*, *308*, *309*
<DTCLOSE>.....111, 113, 286
<DTCOUPON>.....249
<DTCREATED>.....84, *87*, *88*
<DTDUE>*92*, *93*, 102, 129, 130, *170*, *171*, 183, 185, 189, 200, 219, 220, *222*, *223*, *224*, *225*, *226*, *227*, *229*, *230*, *231*, *235*, *236*, B-23
<DTEND>**29**, 30, 105, 107, 110, 111, 112, 113, 165, *166*, 253, 254, *269*, 284, 285, *305*, *306*, *307*, B-17
<DTEXPIRE>70, *71*, 250, *272*, 287, *302*, *305*, *306*, *307*
<DTINFOCHG>.....79
<DTMAT>.....249
<DTNEXT>.....111, 113
<DTOPEN>.....111, 113, 286
<DTPLACED>.....262, *271*
<DTPMTDUE>....31, 113, 286, *305*, *306*, *307*
<DTPMTPRC>189, 192, *223*, *224*, *229*, *235*, B-13
<DTPOSTED>....108, 118, 123, 130, *166*, *270*
<DTPOSTEND>.....111, 112, 113, 114
<DTPOSTSTART>.....111, 112, 113, 114
<DTPRICEASOF>.....264, *270*
<DTPROFUP>20, 61, 62, *166*, *167*, *169*, *171*, *173*, *222*, *223*, *225*, *226*, *228*, *230*, *231*, *232*, *233*, *234*, *269*, *302*, *303*, *310*, B-6, B-21

<DTPURCHASE>.....256, 259, B-16, B-23
<DTPURCHASEDATE>.....B-16
<DTSEEN>.....293
<DTSERVER>20, *166*, *167*, *169*, *170*, *173*, *222*, *223*, *225*, *226*, *228*, *230*, *231*, *232*, *233*, *234*, *269*, *302*, *303*, *304*, *307*, *310*
<DTSETTLE>.....255, *269*
<DTSTART>**29**, 105, 106, 107, 110, 111, 112, 113, 165, *166*, 253, 254, *269*, 284, 285, *304*, *305*, *306*, *307*, *309*
<DTTRADE>.....255, *269*
<DTUPDATE>.....274, 275, *299*, *301*
<DTUSER>.....108, 115, 116, 144, *166*, *270*
<DTXFERPRC>.....103, *174*
<DTXFERPRJ>...118, 123, 130, *168*, *172*, *174*
<DTYIELDASOF>.....250, 251
<DURATION>.....263, *271*
<EMAIL>.....63, 69, *71*, 78, 79, *302*, B-8
<EMAILMSGSET>.....90, B-8
<EMAILMSGSETV1>.....15, 90
<EMAILMSGSETV2>.....15, 90
<EMAILPROF>.....161, 162, **163**, 298
<ENROLLRQ>...18, **69**, *70*, 78, 278, *301*, B-7
<ENROLLRS>..... **70**, *71*, 278, *302*, B-7
<ENROLLTRNRQ>.....69, *70*, 278, *301*
<ENROLLTRNRS>.....70, *71*, *302*
<EVEPHONE>.....69, *71*, 78, 79, 279, *302*
<EXTBANKACCTTO>.....99, **100**, A-2
<EXTBANKDESC>.....128, **129**, 130
<EXTBANKDEST>.....B-10
<EXTDPAYEE>182, **189**, 191, 192, 199, 207, 208, 209, 219, 220, *222*, *224*, *233*, *235*, B-13, B-14, B-16, B-23
<EXTDPMT>.....177, 183, **187**, B-13, B-18
<EXTDPMT2>.....185
<EXTDPMTCHK>.....187, B-13, B-18
<EXTDPMTCHK2>.....187
<EXTDPMTDSC>.....187, B-13, B-18
<EXTDPMTFOR>.....187, B-13, B-18
<EXTDPMTINV>.....187, B-13, B-18
<FAXPHONE>.....62
<FEE>.....116, 130, 144, *169*, B-12
<FEEMSG>.....116, *169*, B-12
<FEES>.....255, 256, 257, 259, 262
<FI>*4*, 19, 20, **21**, *165*, *167*, *168*, *170*, *172*, *221*, *223*, *224*, *226*, *227*, *228*, *229*, *230*, *232*, *233*, *234*, B-3, B-21
<FIASSETCLASS>.....249, 250, 251
<FICERTID>.....22, B-24
<FID>*4*, 21, *165*, *167*, *168*, *170*, *172*, *221*, *223*, *224*, *226*, *227*, *228*, *229*, *230*, *232*, *233*, *234*
<FIID>.....247, 248, *271*, B-15
<FIMFASSETCLASS>.....250
<FINALAMT>.....198, 199, 200, 201
<FINAME>.....62
<FINCHG>.....113
<FINDBILLERRQ>.....274, 299, *301*
<FINDBILLERRS>.....274, **275**, *299*, *301*
<FINDBILLERTRNRQ>.....274
<FINDBILLERTRNRS>.....275, **276**
<FIPORTION>.....250
<FIRSTNAME>.....69, *71*, 78, *301*
<FITID>**27**, 28, 49, 108, 111, 113, *166*, 255, 262, *269*, *270*, *271*, B-22
<FRACCASH>.....256, 259
<FREQ>91, *92*, *93*, *170*, *171*, 200, *229*, *230*, *231*, *236*, A-2, B-8
<FROM>.....84, *87*, *88*
<GAIN>.....256, 257, 258
<GENUSERKEY>.....18, 19
<GETMIME>.....B-8
<GETMIMERQ>.....24, **88**, 89
<GETMIMERS>.....24, **88**, 89, *90*, B-3
<GETMIMESUP>.....90, B-8
<GETMIMETRNRQ>..... *89*
<GETMIMETRNRS>..... *89*
<GROUPID>.....282, 287, 288, 298, *309*
<HASEXTDPMT>.....219, 220
<HEADING>.....277
<HELDINACCT>.....264, *270*
<HELPMESSAGE>.....275, 276, *300*
<HTML>.....*87*, *88*, *90*
<IDSCOPE>.....189, *222*, *224*, *234*, *235*
<IMAGEURL>.....287, *305*, *306*, *307*
<IMG>.....83
<INCBAL>.....253, *269*
<INCIMAGES>84, 86, *87*, *88*, 153, 205, 268, 275, *299*, *301*, B-7, B-8, B-11, B-18
<INCLUDE>.....*4*, 105, 107, *165*, 253, *269*
<INCLUDEDETAIL>.....285, *304*, *307*, *309*
<INCOME>.....258
<INCOMETYPE>.....256, 258, 259
<INCOO>.....253, *269*, B-16
<INCPOS>.....253, *269*, B-16
<INCTRAN>.....*4*, 105, 106, *165*, 253, *269*
<INITIALAMT>.....198, 199, 200, 201
<INTERCANRQ>.....**125**
<INTERCANRS>.....**126**
<INTERMODRQ>.....**124**, B-9
<INTERMODRS>.....**124**
<INTERRQ>.....**122**, 126, 127, 137, 139
<INTERRS>.....**123**, 127, 138, 139, B-9
<INTERSYNCRQ>.....**147**, 151, B-11
<INTERSYNCRS>.....**148**, B-11
<INTERTRNRQ>.....122, 124, 125, 148
<INTERTRNRS>.....123, 124, 126, 148
<INTERXFERMSGSET>.158, 159, 163, B-12
<INTERXFERMSGSETRSV1>.....159
<INTERXFERMSGSETRSV2>.....160
<INTERXFERMSGSETV1> 15, 158, 159, 163
<INTERXFERMSGSETV2> 15, 159, 160, 164
<INTERXFERMSGSRQV1>.....158
<INTERXFERMSGSRQV2>.....159
<INTLXFERFEE>.....164, 165
<INTRACANRQ>.....**121**

<INTRACANRS>.....**121**, *175*
<INTRAMODRQ>.....**119**, 162, B-9
<INTRAMODRS>.....**120**, *174*
<INTRARQ>.....**118**, 133, 134, *167*, *170*
<INTRARS>**118**, 133, 135, *168*, *171*, *173*, *174*, B-9
<INTRASYNCRQ>...**146**, 150, *170*, *172*, B-11
<INTRASYNCRS>.....**147**, *171*, *173*, B-11
<INTRATRNRQ>. *14*, 118, 119, 121, 146, *167*
<INTRATRNRS>118, 120, 121, 147, *167*, *171*, *173*, *174*, *175*
<INVACCTFROM>70, 76, 240, 241, 253, 254, 259, 266, 267, 268, *269*, B-15
<INVACCTINFO>.....240, **241**, B-15
<INVACCTTO>.....76, 241, B-15
<INVACCTTYPE>.....241
<INVALIDACCTTYPE>.....161, B-12
<INVALIDACCTTYPE2>.....162
<INVBAL>.....254, **265**, *270*, B-17
<INVBANKTRAN>.....254, 255, *270*, B-16
<INVBUY>.....**257**, 258, 262, *269*, B-23
<INVDATE>.....187
<INVDESC>.....187
<INVEXPENSE>.....258
<INVMAILRQ>.....**266**
<INVMAILRS>.....**267**
<INVMAILSYNCRQ>.....266, **267**, B-17
<INVMAILSYNCRS>.....266, **268**
<INVMAILTRNRQ>.....268
<INVMAILTRNRS>.....268
<INVNO>.....187
<INVOICE>.....187, 286
<INVOOLIST>.....254, *271*
<INVPAIDAMT>.....187
<INVPOS>.....**264**, 265, *270*, B-17
<INVPOSLIST>.....240, 254, *270*
<INVSELL>.....**257**, 259, 262, B-23
<INVSTMTMSGSET>... **242**, 243, B-15, B-18
<INVSTMTMSGSETV1>....15, 242, 243, 244
<INVSTMTMSGSETV2>....15, 242, 243, 244
<INVSTMTMSGSRQV1>.....243, *268*
<INVSTMTMSGSRQV2>.....243
<INVSTMTMSGSRSV1>.....244, *269*
<INVSTMTMSGSRSV2>.....244
<INVSTMTRQ>.....252, **253**, *268*, B-16
<INVSTMTRS>.....253, **254**, *269*, B-16
<INVSTMTTRNRQ>.....**252**, *268*, B-16
<INVSTMTTRNRS>.....**253**, *269*, B-16
<INVTOTALAMT>.....187
<INVTRAN>.....**255**, 257, 258, 259, *269*, B-16
<INVTRANLIST>.....254, *269*
<ITA.CAUSALE>.....185
<JRNLFUND>.....258
<JRNLSEC>.....258
<LANGUAGE>*4*, 19, 20, 63, *165*, *166*, *167*, *168*, *169*, *170*, *171*, *172*, *173*, *221*, *222*, *223*, *224*, *225*, *226*, *227*, *228*, *229*, *230*, *231*, *232*, *233*, *234*, *268*, *269*, *301*, *302*, *303*, *304*, *306*, *309*, *310*
<LASTNAME>.....69, *71*, 78, 79, *302*
<LEDGERBAL>.....45, 106, 107, *166*
<LIMITPRICE>.....263, *271*
<LINEITEM>.....188
<LITMAMT>.....187, 188
<LITMCODE>.....188, 286
<LITMDESC>.....188
<LOAD>.....255, 256, 257, 259
<LOGO>.....*299*, *300*, *301*, B-7
<LOGO2>.....276
<LOSTSYNC>49, *56*, *57*, 86, 145, 147, 148, 149, 150, 152, 153, 205, 212, 213, 214, 268, B-5, B-18
<MAIL>83, 84, 85, *87*, *88*, 142, 143, 144, 203, 204, 266, 267, 295, B-8
<MAILRQ>.....**85**, *87*
<MAILRS>..... **85**, 86, *87*, *88*
<MAILSUP>.....90, B-8
<MAILSYNCRQ>.....85, **86**, *88*, B-8, B-18
<MAILSYNCRS>.....85, **86**, *88*, B-8, B-18
<MAILTRNRQ>.....85, 86, *87*, B-8
<MAILTRNRS>.....85, 86, *87*, *88*
<MARGINACCT>.....B-17
<MARGINBALANCE>.....265, *270*, B-17
<MARGININTEREST>.....259
<MARKDOWN>.....256, 257
<MARKUP>.....256, 257
<MAX>.....65
<MEMO>**29**, *92*, *93*, 109, 129, 183, 185, *222*, *223*, *224*, *225*, *226*, *227*, *229*, *230*, *231*, *235*, *236*, 249, 255, 263, 264, *270*
<MEMO2>**29**, 103, 109, 129, 185, 249, 255, 263, 264
<MESSAGE>.....13, 26, 70, 80, 85, B-4
<MESSAGE2>.....26
<MFASSETCLASS>.....250
<MFINFO>.....248, **249**, B-15
<MFTYPE>.....249
<MIDDLENAME>.....69, *71*, 78, *301*
<MIN>.....65
<MINAMTDUE>.....286
<MINPMTDUE>.....113
<MINUNITS>.....263
<MKTGINFO>.....106, 107, 112, 114, 254
<MKTVAL>.....264, *270*
<MODELWND>.....162, 219, 220
<MODPENDING>134, 135, 139, 140, 200, 201, *231*, B-10
<MSGBODY>.....84, *87*, *88*
<MSGSETCORE>15, 23, 37, 39, 60, 63, 64, 66, 79, 80, 90, 161, 162, 163, 164, 165, 219, 220, 242, 243, 244, 298, B-6, B-21, B-22, B-24
<MSGSETLIST>.....23, 62, 66, 79, 90
<MULTIINTERTRNRQ>.....126, 148, 164
<MULTIINTERTRNRS>.....**127**, 148

<N>.....290
<NAME>25, 108, 109, 115, 116, 129, 178, 186, 189, 193, 200, *222*, *223*, *224*, *225*, *226*, *227*, *233*, *234*, *235*, *270*, 274, 275, *299*, *300*, *301*, *303*, B-4, B-15, B-17, B-18, B-22
<NAMEACCTHELD>.....279
<NEEDTANPAYEE>.....221
<NEEDTANPMT>.....221
<NEEDTANTRANSFER>.....162
<NEWTAG>.....5
<NEWUNITS>.....256, 259
<NEWUSERPASS>.....21, *23*, 42, 44, B-21
<NINSTS>91, *92*, *93*, 200, *229*, *230*, *231*, *236*, B-8
<NONCE>.....22, B-24
<NOTIFYDESIRED>..286, 293, *305*, *306*, *307*
<NOTIFYWILLING>.....285, 293, *307*, *309*
<NUMERATOR>.....256, 259
<NUMERIC>.....B-6, B-18
<OFX>*3*, *4*, 10, 11, *13*, *14*, 15, 18, 19, 51, 56, 59, 87, *89*, *165*, *166*, *167*, *168*, *169*, *170*, *172*, *173*, *221*, *222*, *223*, *224*, *225*, *226*, *227*, *228*, *229*, *230*, *231*, *232*, *233*, *234*, *268*, *269*, 280, *301*, *302*, *303*, *304*, *306*, *307*, *308*, *309*, B-3
<OFXSEC>.....39, 63
<OLDUNITS>.....256, 259
<ONETIMEPASS>.....19
<OO>.....**262**, 263, 264, *271*, B-16
<OOBUYDEBT>.....263
<OOBUYMF>.....263
<OOBUYOPT>.....263
<OOBUYOTHER>.....263
<OOBUYSTOCK>.....263, *271*
<OODNLD>.....242, 243
<OOSELLDEBT>.....263
<OOSELLMF>.....263
<OOSELLOPT>.....263
<OOSELLOTHER>.....263
<OOSELLSTOCK>.....263
<OOxxxxx>.....254
<OPTACTION>.....256, 258
<OPTBUYTYPE>.....256, 258, 263
<OPTINFO>.....248, **250**, *271*, B-15
<OPTIONALTAG>.....5
<OPTIONLEVEL>.....241
<OPTSELLTYPE>.....256, 259, 263
<OPTTYPE>.....250, *271*
<ORG>*4*, 21, *165*, *167*, *168*, *170*, *172*, *221*, *223*, *224*, *226*, *227*, *228*, *229*, *230*, *232*, *233*, *234*
<ORIGCURRENCY>**45**, 46, 109, 112, 114, 116, 257, 258, 259
<OTHERACCT>.....B-17
<OTHERENROLL>.....79, 80, B-7
<OTHERINFO>.....248, **250**, B-15
<PARVALUE>.....249
<PAYACCT>*92*, *93*, 180, 183, 185, 207, 208, 209, 210, *222*, *223*, *224*, *225*, *226*, *227*, *229*, *230*, *231*, *233*, *234*, *235*, *236*, A-2, B-14, B-16
<PAYACCTS>.....210
<PAYANDCREDIT>.....113
<PAYEE>108, 178, 179, 183, 185, **186**, 192, 193, 200, 207, 208, 209, *222*, *225*, *226*, *227*, *233*, *235*, 294, B-13, B-18
<PAYEE2>.....109, 184, **186**, 207, 209
<PAYEEDELRQ>.....**210**, B-14
<PAYEEDELRS>.....**210**, 211, B-14
<PAYEEID>*92*, *93*, 108, 179, 180, 183, 189, 192, 193, 200, 207, *222*, *223*, *224*, *229*, *230*, *231*, *234*, *235*, *236*, 294, B-14, B-23
<PAYEEID2>.....108, 184, 189, 207, 286
<PAYEELISTID>.....179, 192
<PAYEELSTID>179, 180, 181, 182, 183, 191, 192, 198, 207, 208, 209, 210, 211, *222*, *224*, *230*, *233*, *235*, B-23
<PAYEELSTID2>184, 185, 198, 207, 208, 209, 210, 211
<PAYEEMODRQ>..192, **208**, 210, B-14, B-18
<PAYEEMODRS>..... **209**, B-14, B-18
<PAYEERQ>..181, 192, **206**, *233*, B-14, B-23
<PAYEERS>.....**207**, *233*, B-14
<PAYEESYNCRQ>.....**211**, B-14
<PAYEESYNCRS>.....**212**, B-14
<PAYEESYNRQ>.....182
<PAYEETRNRQ>.....206, 208, 210, 211, *233*
<PAYEETRNRS>182, 207, 209, 210, 212, *233*
<PAYINSTRUCT>.....129, 130
<PAYMENTINSTRUMENT>283, 284, *299*, *300*, *301*
<PAYMENTINSTRUMENTS>276, 283, 284, *299*, *300*, *301*
<PERCENT>.....250
<PHONE>73, *74*, 129, 186, 187, *222*, *225*, *226*, *227*, *233*, *235*, 276, *299*, *300*, *301*, *303*, *304*, B-18
<PINCH>.....65, B-18, B-21, B-22
<PINCHRQ>.....**21**, *23*, 42, 44, A-4, B-3, B-21
<PINCHRS>.....**21**, **22**, *23*
<PINCHTRNRQ>.....21, *23*
<PINCHTRNRS>.....21, *23*
<PMTBYADDR>.....219, 220
<PMTBYPAYEEID>.....219, 220
<PMTBYXFER>.....219, 220
<PMTCANCRQ>.....182, 190, **195**, *227*
<PMTCANCRS>..... *95*, 182, **195**, *228*
<PMTCANRS>.....195
<PMTFOR>.....185
<PMTINFO>*92*, *93*, **183**, 189, 190, 191, 192, 193, 198, 200, 201, 203, 204, 221, *222*, *223*, *224*, *225*, *226*, *227*, *229*, *230*, *231*, *235*, *236*, 294, A-3, B-13, B-23

<PMTINFO2>**183**, 184, 190, 191, 193, 198, 199, 200, 201, 203, 204
<PMTINQRQ>.....182, **196**, *228*
<PMTINQRS>.....**196**, *229*, B-13
<PMTINQTRNRQ>.....196, *228*
<PMTINQTRNRS>.....196, *228*
<PMTINSTRUMENTTYPE>283, **284**, *299*, *300*, *301*
<PMTMAILRQ>.....**203**, B-13
<PMTMAILRS>.....**204**, B-13
<PMTMAILSYNCRQ>.....**204**, B-13
<PMTMAILSYNCRS>.....**205**, B-13
<PMTMAILTRNRQ>.....203, 205
<PMTMAILTRNRS>.....204, 205
<PMTMODRQ>177, 181, 190, **193**, *224*, *226*, B-13
<PMTMODRS> 182, **193**, 195, *225*, *227*, B-13
<PMTMSGSET>.....B-14
<PMTMSGSETV1>.....B-14
<PMTPRCCODE>189, **192**, *223*, *224*, *229*, *235*, B-13
<PMTPRCSTS>**189**, 191, 193, 196, *223*, *224*, *229*, *235*, B-13
<PMTRQ>*57*, 180, 181, 182, **190**, 191, 192, 195, 206, 207, 209, *221*, *223*
<PMTRS>*57*, 181, 182, **190**, 191, 192, *222*, *224*, *235*, B-13, B-23
<PMTSYNCRQ>51, *56*, *57*, *94*, 182, **212**, 215, *234*, B-14
<PMTSYNCRS>*56*, *57*, *95*, 182, **213**, 215, *235*, B-14
<PMTTRNRQ>51, *57*, 190, 193, 195, 213, *221*, *223*, *224*, *226*, *227*, B-14
<PMTTRNRS>*57*, *95*, 191, 193, 195, 213, *222*, *224*, *225*, *226*, *228*, *235*
<PMTTYPE>.....184, A-3
<PORTION>.....250
<POSDEBT>.....265
<POSDNLD>.....242, 243
<POSMF>.....265
<POSOPT>.....265, *270*
<POSOTHER>.....240, 265
<POSSTOCK>.....265, *270*
<POSTALCODE>62, 69, *71*, 78, 79, 129, 186, *222*, *225*, *226*, *227*, *233*, *235*, 275, 276, 279, *299*, *300*, *301*, *302*, *303*, *304*, B-6, B-18
<POSTPROCWND>.....219, 220
<POSTYPE>.....256, 259, 264, *270*
<POSxxxxx>.....254
<PREAUTH>.....79, 80
<PREAUTHTOKEN>.....76, 80, 280
<PREFECTCCHURL>.....287
<PREFETCHURL>.....287
<PREFIX>.....64
<PRESACCTFROM>275, 278, **279**, 286, 289, 293, 295, *305*, *306*, *307*, *310*
<PRESACCTINFO>...278, 281, 282, 294, *310*
<PRESACCTINFORS>.....*310*
<PRESACCTTO>.....275, **279**, 280, *303*
<PRESBILLINFO>285, **286**, 289, 293, 294, *305*, *306*, *307*
<PRESDELIVERYID>.....293, 294
<PRESDETAIL>.....287, 289
<PRESDETAILRQ>.....284, 286, 289
<PRESDETAILRS>.....286, **289**, *305*, *307*
<PRESDETAILTRNRQ>.....289
<PRESDETAILTRNRS>.....289, **291**
<PRESDIRMSGSET>.....296, 298
<PRESDIRMSGSETV1>....273, 274, 296, 298
<PRESDIRMSGSETV21>.....15
<PRESDIRMSGSRQV1>.....274, 296
<PRESDIRMSGSRSV1>.....274, 296
<PRESDIRPROF>.....298
<PRESDLVMSGSET>.....296, 297, 298
<PRESDLVMSGSETV1>...273, 284, 297, 298
<PRESDLVMSGSRQV1>284, 297, *304*, *306*, *309*
<PRESDLVMSGSRSV1>284, 297, *304*, *307*, *310*
<PRESDLVPROF>.....298
<PRESDVLMSGSETV21>.....15
<PRESGRPACCTINFOTRNRQ>281, **282**, *309*
<PRESGRPACCTINFOTRNRS>281, **282**, **283**, *310*
<PRESLIST>.....285, *305*, *307*
<PRESLISTRQ>284, 285, 286, 287, 288, 293, 294, *304*, *306*, *309*
<PRESLISTRS>.....284, **285**, 288, *305*, *307*
<PRESLISTTRNRQ>284, **287**, 288, *304*, *306*, *309*
<PRESLISTTRNRS>..285, 287, **288**, *304*, *307*
<PRESMAILRQ>.....**295**
<PRESMAILRS>.....**295**
<PRESMAILSYNCRQ>.....294
<PRESMAILSYNCRS>.....294
<PRESMAILTRNRQ>.....295
<PRESMAILTRNRS>.....**295**
<PRESNAMEADDRESS>.....279, 280, *303*
<PRESNOTIFYRQ>.....**293**
<PRESNOTIFYRS>.....**294**
<PRESNOTIFYTRNRQ>.....293
<PRESNOTIFYTRNRS>.....294
<PREVBAL>.....286, 287
<PROCDAYSOFF>162, 163, 164, 165, 219, 220, 298, B-12, B-14
<PROCENDTM>162, 163, 164, 165, 219, 220, 298
<PROFMSGSET>.....66
<PROFMSGSETV1>.....15, 66
<PROFMSGSETV2>.....15, 66
<PROFMSGSV1>.....60
<PROFMSGSV2>.....60
<PROFRQ>.....18, **61**, B-6
<PROFRS>**62**, 296, 298, B-6, B-18, B-21, B-22, B-24

<PROFTRNRQ>.....61
<PROFTRNRS>.....62
<PURANDADV>.....113
<PWTYPE>.....19, 65
<RATING>.....248
<REASON>.....278, *310*
<RECINTERCANRQ>.....**141**
<RECINTERCANRS>.....**141**
<RECINTERMODRQ>.....**139**, B-10
<RECINTERMODRS>.....**139**
<RECINTERRQ>.....**137**
<RECINTERRS>.....**137**, 138
<RECINTERSYNCRQ>.....**151**, B-11
<RECINTERSYNCRS>.....**152**, B-11
<RECINTERTRNRQ>.....137, 139, 141, 152
<RECINTERTRNRS>.....137, 139, 141, 152
<RECINTRACANRQ>.....**136**, *172*
<RECINTRACANRS>.....**136**, *173*
<RECINTRAMODRQ>.....**134**, 162
<RECINTRAMODRS>.....**134**
<RECINTRARQ>.....**132**, *170*
<RECINTRARS>.....**133**, *171*
<RECINTRASYNCRQ>. **150**, *172*, B-11, B-24
<RECINTRASYNCRS>... **150**, 151, *173*, B-11
<RECINTRATRANRQ>.....B-22
<RECINTRATRNRQ>132, 134, 136, 150, *170*, *172*
<RECINTRATRNRS>133, 134, 136, 151, *171*, *173*
<RECPMTCANCRQ>.....*94*, **202**, *232*, B-13
<RECPMTCANCRS>.. *94*, *95*, **202**, *232*, B-13
<RECPMTCANRQ>.....202
<RECPMTMODRQ>.....**200**, *230*, B-13
<RECPMTMODRS>.....**201**, *231*, B-13
<RECPMTRQ>.....*92*, **198**, *229*, B-13
<RECPMTRS>. *93*, **198**, *230*, *235*, B-13, B-23
<RECPMTSYNCRQ>.....**213**, 215, *234*, B-14
<RECPMTSYNCRS>.....**214**, 215, *235*, B-14
<RECPMTTRNRQ>198, 200, 202, 214, *229*, *230*, *232*, B-14
<RECPMTTRNRS>198, 201, 202, 214, *230*, *231*, *232*, *235*
<RECSRVRTID>92, *93*, 94, *95*, 118, 123, 133, 134, 136, 138, 139, 141, *171*, *172*, *173*, *174*, 180, 191, 198, 200, 201, 202, *230*, *231*, *232*, *235*
<RECSRVRTID2>119, 123, 133, 134, 135, 136, 138, 139, 141, 191, 198, 200, 201, 202
<RECURRINST>**91**, *92*, *93*, 132, 133, 134, 135, 137, 138, 139, *170*, *171*, 198, 200, 201, *229*, *230*, *231*, *236*, B-8
<REFNUM>.....108, 123
<REFRESH>52, 86, 145, 146, 147, 149, 150, 151, 152, 204, 205, 211, 212, 213, 214, 267, B-5, B-8, B-9, B-11, B-14, B-17, B-18
<REINVCG>.....265
<REINVDIV>.....265, B-18
<REINVEST>.....259
<REJECTIFMISSING>51, 52, *56*, *57*, 86, *88*, *94*, 145, 146, 147, 149, 150, 151, 153, *170*, *172*, 205, 211, 213, 214, *234*, 267, B-5, B-8, B-11, B-12, B-14, B-17, B-18
<RELFITID>.....256, 258, 259
<RELTYPE>.....256, 259
<REQUIREDTAG>.....5
<REQUIREDTAG2>.....5
<RESPFILEER>.....64, B-21, B-22
<RESTRICT>.....276, *300*
<RESTRICTION>.....263, *271*
<RETOFCAP>.....259
<REVERSAL>.....257, 262
<REVERSALAMTFEES>.....257
<REVERSALFEE>.....262
<REVERSALFEES>.....256, 262
<REVERSALFITID>.....256, 257, 262
<SCVCADD>.....74, 75
<SECID>**246**, 247, 248, 250, 257, 258, 259, 262, 264, *270*, *271*, *272*, B-15
<SECINFO>**248**, 249, 250, 251, *271*, B-15, B-17, B-18
<SECLIST>.....248, *271*
<SECLISTMSGSET>.....**244**, 245
<SECLISTMSGSETV1>.....15, 244, 245
<SECLISTMSGSETV2>.....15, 244, 245
<SECLISTMSGSRQV1>.....245
<SECLISTMSGSRQV2>.....245
<SECLISTMSGSRSV1>.....245, *271*
<SECLISTMSGSRSV2>.....245
<SECLISTRQ>.....246, 247, 248, B-15
<SECLISTRQDNLD>.....244
<SECLISTRS>.....247, **248**, B-23
<SECLISTTRNRQ>.....**246**, B-15
<SECLISTTRNRS>.....**247**, B-15
<SECNAME>.....248, *271*, B-15, B-17, B-18
<SECRQ>.....247, B-18
<SECURED>.....256, 259, 265
<SECURITY>.....43
<SECURITYNAME>.....70, *71*, *302*
<SELLALL>.....263
<SELLDEBT>.....259
<SELLMF>.....259
<SELLOPT>.....259, B-16
<SELLOTHER>.....240, 259
<SELLREASON>.....256, 259
<SELLSTOCK>.....259
<SELLTYPE>.....256, 259, 263
<SESSCOOKIE>.....18, 20, B-3
<SEVERITY>13, 16, *23*, 26, *71*, *74*, *77*, *87*, *88*, *89*, *95*, *166*, *167*, *168*, *169*, *170*, *171*, *173*, *174*, *175*, *222*, *223*, *224*, *225*, *226*, *227*, *228*, *229*, *230*, *231*, *232*, *233*, *234*, *235*, *269*, *302*, *303*, *304*, *307*, *310*
<SHORTACCT>.....B-17
<SHORTBALANCE>.....265, *270*, B-17
<SHPERCTRCT>.....250, 256, 258, 259, *272*
<SIC>.....108, 275, 276, *299*, *300*, *301*

<SIGNON>.....56
<SIGNONINFO>62, 65, B-6, B-18, B-22, B-24
<SIGNONINFOLIST>.....62
<SIGNONMSGSET>.....23, B-3, B-21
<SIGNONMSGSETV1>.....15, 23, 60, B-24
<SIGNONMSGSETV2>.....15, 23
<SIGNONMSGSRQV1>*4*, *14*, 18, *165*, *167*, *168*, *169*, *172*, *221*, *223*, *224*, *226*, *227*, *228*, *229*, *230*, *232*, *233*, *234*, *268*, B-24
<SIGNONMSGSRQV2>*301*, *302*, *304*, *306*, *308*, *309*
<SIGNONMSGSRSV1>18, *166*, *167*, *169*, *170*, *173*, *222*, *223*, *225*, *226*, *227*, *228*, *229*, *231*, *232*, *233*, *234*, *269*, B-24
<SIGNONMSGSRSV2>*302*, *303*, *304*, *307*, *309*
<SIGNONREALM>.....63, 65
<SIGNUPMSGSET>.....79, B-7
<SIGNUPMSGSETV1>.....15, 79
<SIGNUPMSGSETV2>.....15, 80
<SIGNUPMSGSRQV1>.....B-18
<SIGNUPMSGSRQV2>.....*301*, *303*
<SIGNUPMSGSRSV1>.....B-18
<SIGNUPMSGSRSV2>.....*302*, *303*
<SIGNUPMSGSV1>.....67
<SIGNUPMSGSV2>.....67
<SONRQ>*4*, *14*, **18**, **19**, 20, 38, 42, 43, 61, 69, 87, *165*, *167*, *168*, *169*, *172*, *221*, *223*, *224*, *226*, *227*, *228*, *229*, *230*, *232*, *233*, *234*, *268*, 280, 282, 287, 288, *301*, *302*, *304*, *306*, *308*, *309*, A-4, B-3, B-21
<SONRS>**18**, 20, *166*, *167*, *169*, *170*, *173*, *222*, *223*, *225*, *226*, *227*, *228*, *229*, *231*, *232*, *233*, *234*, *269*, *302*, *303*, *304*, *307*, *309*, B-3, B-21, B-24
<SOURCE>.....B-15
<SPACES>.....65
<SPECIAL>.....65
<SPECIFIC>.....5
<SPLIT>.....259
<SPNAME>.....20, 64, B-21, B-22
<SRVRTID>**28**, 49, 50, 51, 52, 54, *95*, 108, 118, 119, 120, 121, 123, 124, 125, 126, 130, 131, 133, *168*, *171*, *173*, *174*, *175*, 180, 181, 182, 191, 193, 195, 196, 203, 204, 215, 216, *222*, *224*, *225*, *226*, *227*, *228*, *229*, *235*, 255, 262
<SRVRTID2>**28**, 108, 118, 120, 121, 123, 124, 125, 126, 130, 131, 191, 193, 195, 196, 203, 204, 255, 262
<STATE>62, 69, *71*, 78, 79, 129, 186, *222*, *225*, *226*, *227*, *233*, *235*, 275, 276, 279, *299*, *300*, *301*, *302*, *303*, *304*
<STATUS>*13*, 16, 17, 20, 21, 22, *23*, **26**, 49, 52, *57*, 62, 70, *71*, *74*, *77*, *87*, *88*, *89*, *95*, 127, *166*, *167*, *169*, *170*, *171*, *173*, *174*, *175*, *222*, *223*, *224*, *225*, *226*, *227*, *228*, *229*, *230*, *231*, *232*, *233*, *234*, *235*, 247, 253, *269*, 277, 278, 282, 288, *302*, *303*, *304*, *307*, *309*, *310*, B-2, B-4, B-5, B-15, B-16, B-20, B-21
<STMNTIMAGE>.....286, **287**, *305*, *306*, *307*
<STMTENDRQ>.....110, 112
<STMTENDRS>.....110, **111**, 112, B-9
<STMTENDTRNRQ>.....110
<STMTENDTRNRS>.....111
<STMTRQ>*4*, 13, 105, 106, 111, 112, *165*, 240, B-9
<STMTRS>....13, 45, 105, 107, *166*, 240, B-9
<STMTTRN>105, 106, 107, **108**, *166*, 255, *270*, 290, B-9, B-22
<STMTTRNRQ>.....*4*, *14*, 105, *165*
<STMTTRNRS>.....105, *166*
<STOCKINFO>.....248, **251**, *271*, B-15
<STOCKTYPE>.....251
<STOPCHKPROF>.....B-18
<STOPPRICE>.....263
<STPCHKFEE>.....163
<STPCHKNUM>.....116, *169*, B-9
<STPCHKPROF>.....161, 162, **163**, B-12
<STPCHKRQ>.....**115**, *168*, A-2, B-9
<STPCHKRS>.....**116**, *169*, B-9
<STPCHKSYNCRQ>.....**145**, B-9, B-11
<STPCHKSYNCRS>.....**145**, B-11
<STPCHKTRNRQ>.....115, 145, *168*
<STPCHKTRNRS>.....116, 146, *169*
<STRIKEPRICE>.....250, *272*
<STSVIAMODS>.....219, 220
<SUBACCT>.....263, *271*
<SUBACCTFROM>.....256, 258
<SUBACCTFUND>255, 256, 257, 258, 259, *270*
<SUBACCTSEC>.....256, 257, 258, 259, *270*
<SUBACCTTO>.....256, 258
<SUBJECT>.....84, *87*, *88*
<SUPPORTDTAVAIL>.....162, 185, 221
<SUPTXDL>..... *74*, 101, 102
<SVC>.....75, *77*, 280, *303*, *304*, A-3
<SVC2>.....75
<SVCADD>74, 75, **76**, *77*, 280, 281, *303*, B-7, B-18
<SVCCHG>.....74, 75, **76**, B-7, B-18
<SVCDEL>.....74, 75, **76**, B-7
<SVCSTATUS>72, 73, *74*, 75, *77*, 101, 102, 183, 241, 278, 281, *310*, B-7, B-13, B-18
<SWITCHALL>.....264
<SWITCHMF>.....264
<SYNCERROR>49, 52, 86, 146, 147, 148, 149, 151, 152, 153, 205, 212, 213, 214, 268
<SYNCMODE>.....64, B-21, B-22
<TABLENAME>.....289, *305*, *307*, *308*
<TABLETYPE>.....**290**, 292
<TAN>16, 126, 221, 246, 252, 282, 288, B-15, B-16, B-18
<TAXES>.....255, 257, 259
<TAXEXEMPT>.....257, 258, 259, B-18
<TAXID>.....25, 70, *71*, *302*, B-4

<TBLSTRUCTRQ>.....**292**
<TBLSTRUCTRS>.....**293**
<TEMPPASS>.....70, *71*, *302*
<TFERACTION>.....257, 259
<TICKER>.....247, 248, *271*, B-15
<TO>.....84, *87*, *88*
<TOKEN>**28**, 49, 50, 51, 52, *56*, *57*, 77, 86, 88, *94*, *95*, 143, 145, 146, 147, 148, 149, 150, 151, 152, 153, *170*, *171*, *172*, *173*, 204, 205, 211, 212, 213, 214, 216, *234*, *235*, 267, 268, B-5, B-8, B-12, B-14
<TOKEN2>**28**, 52, 86, 145, 146, 147, 148, 149, 150, 151, 152, 153, 204, 205, 211, 212, 213, 214, 267, 268
<TOKENONLY>52, 86, 145, 146, 147, 149, 150, 151, 152, 153, 204, 205, 211, 212, 213, 214, 267, B-5, B-8, B-9, B-11, B-14, B-17, B-18
<TOTAL>.....256, 257, 258, 259, *270*
<TOTALFEES>.....112
<TOTALINT>.....112
<TRANDNLD>.....242, 243
<TRANSFER>.....259, B-16, B-23, B-24
<TRANSPSEC>.....37, 38, 63, 64, B-22
<TRNAMT>**28**, 29, *92*, *93*, 102, 108, 115, 116, 129, 130, 144, *166*, *167*, *168*, *170*, *171*, *174*, 183, 184, *222*, *223*, *224*, *225*, *226*, *227*, *229*, *230*, *231*, *235*, *236*, *270*
<TRNRS>.....53
<TRNTYPE>.....108, **109**, *166*, *270*
<TRNUID>*4*, 13, 16, *23*, **28**, 49, 52, 54, 55, *57*, *70*, *71*, *73*, *74*, *77*, 83, 85, *87*, *88*, *89*, *95*, 126, 127, *165*, *166*, *167*, *168*, *169*, *170*, *171*, *172*, *173*, *174*, *175*, 180, 215, 216, *221*, *222*, *223*, *224*, *225*, *226*, *227*, *228*, *229*, *230*, *231*, *232*, *233*, *235*, 246, 247, 252, 253, *268*, *269*, 282, 288, *301*, *302*, *303*, *304*, *306*, *307*, *309*, *310*, A-2, B-2, B-4, B-5
<TSKEYEXPIRE>.....20
<TSPHONE>.....62
<TYPEDESC>.....251
<UNIQUEID>.....246, *270*, *271*, *272*
<UNIQUEIDTYPE>.....246, *270*, *271*, *272*
<UNITPRICE>240, 248, 257, 259, 264, *270*, B-16, B-23
<UNITS>240, 257, 258, 259, 262, 263, 264, *270*, *271*
<UNITSSTREET>.....265
<UNITSUSER>.....265
<UNITTYPE>.....257, 263, 264
<URL>.....*24*, 60, 62, 63, 64, 80, 89, *90*
<URL2>.....62, 63, 80, 89
<URLGETREDIRECT>.....62
<USEHTML>84, 85, 86, *87*, *88*, 153, 205, 268, B-8, B-11, B-18
<USERID>*4*, 18, 19, 21, 22, *23*, 61, 68, 69, 70, *71*, 78, 84, *87*, *88*, *165*, *167*, *168*, *170*, *172*, *221*, *223*, *224*, *226*, *227*, *228*, *229*, *230*, *232*, *233*, *234*, *268*, 279, 280, 282, 285, 287, 288, *301*, *302*, *303*, *304*, *305*, *306*, *307*, *309*, *310*, B-3
<USERKEY>.....18, 19, 20, B-3
<USERPASS>*4*, 18, 19, 42, 43, 61, 68, *165*, *167*, *168*, *170*, *172*, *221*, *223*, *224*, *226*, *227*, *228*, *229*, *230*, *232*, *233*, *234*, *268*, *301*, *303*, *304*, *306*, *309*, A-3, B-3, B-21
<USPRODUCTTYPE>.....46, 241, 242
<VALIDATE>.....275, 276, 277, *300*
<VALIDATE2>.....276
<VALUE>.....25, *271*
<VER>.....63, 64, B-22
<WEBENROLL>.....79, 80, B-7
<WIREBENEFICIARY>...128, **129**, 130, B-10
<WIRECANRQ>.....**131**
<WIRECANRS>.....**131**
<WIREDESTBANK>.....128, 130, B-10
<WIRERQ>.....**128**, B-10
<WIRERS>.....**130**, B-10
<WIRESYNCRQ>.....**149**, B-11
<WIRESYNCRS>.....**149**, B-11
<WIRETRNRQ>.....128, 131, 149
<WIRETRNRS>.....130, 131, 149
<WIREXFERMSGSET>...160, 161, 164, B-12
<WIREXFERMSGSETV1>15, 154, 160, 161, 164
<WIREXFERMSGSETV2>..15, 160, 161, 165
<WIREXFERMSGSRQV1>.....160
<WIREXFERMSGSRQV2>.....160
<WIREXFERMSGSRSV1>.....161
<WIREXFERMSGSRSV2>.....161
<WITHHOLDING>.....257, 258
<XFERDAYSWITH>.....219, 220
<XFERDEST>..... *74*, 101, 102
<XFERDFLTDAYSTOPAY>.....185, 219, 220
<XFERINFO>**102**, 118, 119, 120, 122, 123, 124, 133, 134, 135, 137, 138, 139, 162, *167*, *168*, *170*, *171*, *173*, *174*, B-9
<XFERPRCCODE>.....103, *174*
<XFERPRCSTS>**103**, 119, 120, 123, 125, *174*, B-9
<XFERPROF>.....161, 162, 164, B-12, B-18
<XFERSRC>..... *74*, 101, 102
<XXXACCTFROM>.....70, 73, 74, 76, B-7
<XXXACCTINFO>.....73
<XXXACCTTO>.....25, 76, B-7
<XXXINFO>.....248
<XXXMSGSET>.....62, 63
<XXXMSGSETVn>.....63
<XXXMSGSRQVn>.....14, B-2
<XXXMSGSRSVn>.....14, B-2
<XXXMSGSVn>..... *13*
<XXXRQ>.....13
<XXXRS>.....13, B-18
<XXXSYNCRQ>.....14, B-18
<XXXSYNCRS>.....14, B-18
<XXXTRNRQ>.....14, 16, 28

<XXXTRNRS>.....14, 16, B-18
<XYZ>.....12
<YIELD>.....249, 251, *271*
<YIELDTOCALL>.....249
<YIELDTOMAT>.....249
<YYYSYNCRQ>..... *13*
<YYYSYNCRS>..... *13*
<YYYTRNRQ>..... *13*
<YYYTRNRS>..... *13*