Functional Elements Requirements

Working Draft 01a, 1-Jul-2004

Document identifier:

FWSI-FESC-Requirements-01a.doc

Location:

http://www.oasis-open.org/apps/org/workgroup/fwsi/documents.php

Editor:

Tan Puay Siew, Singapore Institute of Manufacturing Technology, SIMTech (pstan@simtech.a-star.edu.sg)

Contributors:

Ang Chai Hong, SIMTech (chang@simtech.a-star.edu.sg)
Chan Lai Peng, SIMTech (lpchan@simtech.a-star.edu.sg)
Cheng Jason, SIMTech (jason@simtech.a-star.edu.sg)
Cheng Yushi, SIMTech (ycheng@simtech.a-star.edu.sg)
Dilip Kumar Limbu, SIMTech (dilip@simtech.a-star.edu.sg)
Wu Yingzi, SIMTech (yzwu@simtech.a-star.edu.sg)
Xu Xingjian, SIMTech (xjxu@simtech.a-star.edu.sg)

Abstract:

The ability to provide robust implementations is a very important aspect to create high quality Web Service-enabled applications and to accelerate the adoption of Web Services. The Framework for Web Services Implementation (FWSI) TC aims to enable robust implementations by defining a practical and extensible methodology consisting of implementation processes and common functional elements that practitioners can adopt to create high quality Web Services systems without reinventing them for each implementation.

This document serves as a supporting document towards the identification of common functional elements, which in turn will be detailed in the Functional Elements Specification. In this document, aspects pertaining to enabling a robust Web Service-enabled application are discussed and the functional requirements arising out of these aspects are detailed.

Status:

This document is updated periodically on no particular schedule.

Committee members should send comments on this specification to the fwsi@lists.oasis-open.org list. Others should subscribe to and send comments to the fwsi-comment@lists.oasis-open.org list. To subscribe, send an email message to fwsi-comment-request@lists.oasis-open.org with the word "subscribe" as the body of the message.

For information on whether any patents[1] have been disclosed that may be essential to implementing this specification, and any offers of patent licensing terms, please refer to the Intellectual Property Rights section of the FWSI TC web page (http://www.oasis-open.org/committees/fwsi/).

[1] This document contains concepts that have been filed as patents. The Intellectual Property Rights declaration and contractual terms on use of document's content will be made available at a later date.

Table of Contents

1 Introduction

The purpose of OASIS Framework for Web Services Implementation (FWSI) Technical Committee (TC) is to facilitate implementation of robust Web Services by defining a practical and extensible methodology consisting of implementation processes and common functional elements that practitioners can adopt to create high quality Web Services systems without re-inventing them for each implementation. It aims to solve the problem of the slow adoption of Web Services due to a lack of good Web Services methodologies for implementation, cum a lack of understanding and confidence in solutions that have the necessary components to reliably implement Web Service-enabled applications.

One of the FWSI TC's deliverables is the Functional Elements Specification. This Specification specifies a set of functional elements that practical implementation of Web Services-based systems will require. A Functional Element (FE) is defined as a building block representing common reusable functionalities for Web Service-enabled implementations, i.e. from an application Point-Of-View. These FEs are expected to be implemented as reusable components, with Web Services capabilities where appropriate, and to be the foundation for practitioners to instantiate into a technical architecture. The implementations of these FEs are further supported by another complementary work that is also from the FWSI TC, the Web Services Implementation Methodology (WSIM) [1]. As such, the TC hopes that through the implementations of these FEs, robust Web Service-enabled applications can be constructed quickly and deployed in a rapid manner.

This document serves as a supporting document towards the identification of common functional elements, which in turn will be detailed in the Functional Elements Specification. It discusses the aspects pertaining to enabling a robust web service-enabled application from an application Point-Of-View and also detailed the functional requirements arising out of these aspects. Presently, the requirements are categorised into four main areas; namely Management, Process, Delivery and Security.

The target audiences for this document are expected to be solution providers who intend to use the Functional Elements Specification to create building blocks that can be instantiated into the technical architecture of their solutions or software vendors and independent software vendors (ISVs) that are expected to build the functional elements specified into their products. Individuals and researchers who are interested in Web Services will also be able to benefit from this document. It is recommended that this document should be used in tandem with the Functional Elements Specification, to ensure that readers have a holistic view to the thought processes and knowledge that are encapsulated.

1.1 Scope

As the FEs are to be identified from an application Point-Of-View, correspondingly, the requirements are also identified from the same viewpoint, i.e. application based. This document covers only requirements that are considered to be functional in nature as it is targeted to be a reference base point for the Functional Elements Specification. Furthermore, the Specification is not expected to cover implementation details of individual FEs. Thus non-functional requirements arising from aspects like Usability, Reliability, Performance, Supportability and any other Design Constraints are not articulated in this document.

The first three categories, i.e. Management, Process and Delivery mentioned in the previous section handle aspects that are important to a Web Service-enabled application and the fourth category Security, handles the security related requirements arising from the first three.

1.2 Glossary, Acronyms and Abbreviations

Assertion Authority	An entity within a COT that provides authentication assertions.
Channel	A logical grouping of event consumers. When an event is routed to the channel, all the event consumers who subscribed to this channel will receive the notification of the event.
cHTML	Compact HTML
COT	Circle Of Trust
DB	Database
ebXML	Electronic Business XML
Event	An Event is defined as an activity that occurs in a business process to indicate the status of an action and normally it triggers one or more follow-up actions. An Event is identified by name and associated with a set of attributes. An Event occurs within a certain timeframe and it can be identified through inspecting the changes in its associated data. The same Event may re-occur during the said business process.
Event Consumer	A receiver of a set of action(s) that was triggered by an Event Supplier. An Event Consumer can be a person, an application or a service.
Event Supplier	An event generator or an event trigger. An Event Suppler can be a person, an application or a service.
FE	Functional Element
Filter	A condition defined by an event consumer for filtering out unwanted or unsolicited events or events' notification. Routing rules are used to filter undesired events from reaching targeted Event Consumers.
FWSI	Framework for Web Services Implementation
Group	A Group is a collection of individual users, and are typically grouped together as they have certain commonalities.
HTTP	HyperText Transfer Protocol
IDP	Identity Provider manages user's identity profile within a COT. Sometimes an IDP may function as an Assertion Authority too.
IT	Information Technology
Key Agreement	A public-key cryptography protocol that allows two parties to establish a shared secret over an unsecured communications channel. For example using Diffie-Hellman to establish session keys.
LAN	Local Area Network
LDAP	Lightweight Directory Access Protocol
MIME	Multipurpose Internet Mail Extension
PDA	Personal Digital Assistant
PKI	Public Key Infrastructure
RDBMS	Relational Database Management System
Resource	A resource in an application is defined to encompass users, services, data / information, transaction and security.
Role	A role is typically assigned to a user to define or indicate the job or responsibility of the said user in a particular context.
Routing Rule	A routing rule is an expression specifying where (to either channel(s) or event consumers) events are routed. A routing rule is associated with an event.
SAML	Security Assertions Markup Language
SMS	Short Message Service
SOA	Service-oriented Architecture A Service-oriented Architecture is essentially a collection of services that communicate with each other. The communication can involve either simple data passing or services coordinating some activities [2].
SOAP	Simple Object Access Protocol
SOS	Single Sign-On
SSL/TLS	Secured Sockets Layer / Transport Layer Security
TC	Technical Committee
User	A user is loosely defined to include both human and virtual users. Virtual

	users could include service users and application (or machine) users that are utilising other services in a SOA environment.
WAP	Wireless Application Protocol

120

2 Requirements

As briefly mentioned in section 1.2, the functional requirements arising out of a Web Service-enabled application are categorised into Management, Process, Delivery and Security. The *Management* category handles requirements that arise from the management of an application, in particular an application that is designed using the SOA model and implemented in a distributed environment. This includes the management of resources and its access or utilisation in a Web Service-enabled application. The *Process* category handles aspects that are related to enabling the execution of a sequence of tasks, which include handling requirements arising out of executing these tasks in a distributed environment that typically comprises more than one interacting services.

On the other hand, *Delivery* aims to enable applications to handle a myriad of access mechanisms provided through different devices, bandwidth availability, and input cum output formats supported. The last category, *Security,* relates to requirements arising from the need to ensure that an application is secure and not prone to hacks or unwanted intrusion that could cause untold damages once deployed. Some aspects that are important here include the need to ensure sensitive information is secured during transit and safely guarded at its source and destination.

2.1 Management

Under the *Management* aspects, there is a need to handle two major areas; *Management of Resources* and *Management of Access to Resources* [3]. A *Resource* in an application is defined to encompass users, services, data / information, transaction[2] and security. As such *Management* includes all these aspects. In any application, the ability to manage resources is of utmost importance, and all information and activities pertaining to these resources must be tracked and managed to ensure system integrity and information correctness. It is also important to capture pertinent user's information. One example is user preferences. Through capturing such information and tracking the users' usage or utilisation pattern, it is possible to provide useful personalisation services to enhance the entire user's experience of an application. This task can be very complex and demanding, especially for applications that are built based on the SOA model, where these resources can span multiple applications or services on different platforms that are distributed throughout the enterprise or even across enterprises.

For the rest of this section, an illustration of *Resources* is exemplified using the concept of users. Here, a *User* is used in a generic manner, to include a human user, a service user and even an application or a machine that is utilising another service. It is expected that throughout the lifecycle of an application, the users' profile will change, to accommodate the addition of new users, deletion of inactive of ineligible users, or even modification of users' information. Furthermore, a particular user's role(s) is bound to change, either through the passage of time or when a human user participates in different projects or assignments in different capacities. As such, users are typically assigned *Role(s)* in an application. A *Role* is defined as the responsibility or function of that user in a said context.

Next, individual users could also be grouped into *Groups*. A *Group* is a collection of individual users, and are typically grouped together as they have certain commonalities. Examples of such commonalities include staff of a department, users of a particular type of service or team members of a project. The purpose of *Groups* is usually to ease the task of managing resources (such as *Users* in this case) and reducing the complexity of assigning correct access to the application resources. As such, typically, members of a group could be assigned same *Role(s)* also.

[2] Transaction aspects of *Management of Resources* are to be detailed in later version(s).

For any given time, a user/group could have multiple *Roles*, each with an identified set of responsibilities in a specific context, and specified accessibility to other *Resources* within the application. For example, a human user (*User*) could be assigned as an "Administrator" for a department-xyz and as a "Normal User" of the company. As such as the "Administrator" of department-xyz, this *User* can now add, delete and modify account information pertaining to her department. However, she cannot do likewise for other departments as her role is only as a "Normal User". Through the assignment of roles, a said user's access to other *Resources* can be controlled. The same is applicable for a *Role* assigned to a *Group*, in this case, all members of the group will have the same assigned *Role(s)* and same access to specified *Resources*.

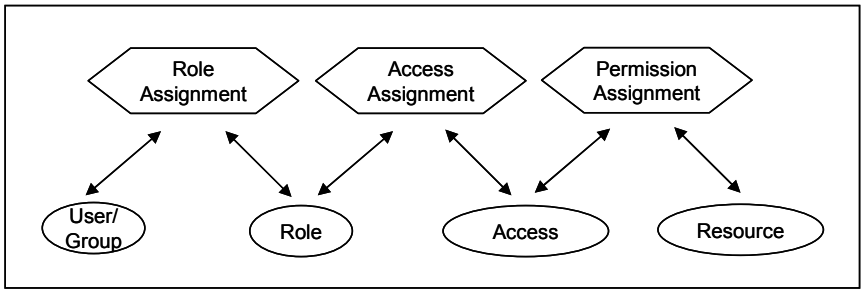

Figure 1: Indirect Assignment of Access to Resources through the Concept of Role

Figure 1 illustrates the concepts discussed about *User, Group, Role* and *Resource* in a graphical manner. By decoupling users' access to resources through the concept of *Role*, an application developed and deployed in this manner is much more flexible, and able to operate better in a SOA model, where services and users are expected to be across multiple domains (organisations) and an application can expect users to be from different domains also. This brings about the need to manage a user's identity across multiple domains, or more commonly known as enabling Single-Sign-On (SSO). This can be further enhanced through another concept called Federated Identity. Both the SSO and Federated Identity concepts are particular useful for managing users' identity in terms of providing a means to specify the users' identity, means to prove the users' identity, and as a means to manage and protect the users' identity. The security aspects of this identity management are covered in section 2.4.

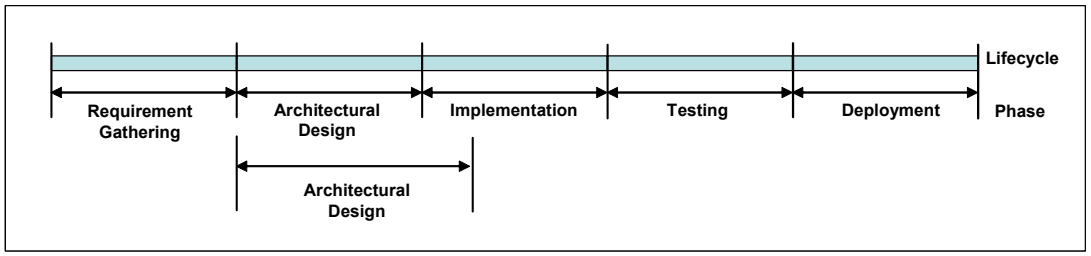

Figure 2: An example of the relationships between Phases and a Lifecycle of a Project

There is another concept that needs to be managed, namely the influence of temporal aspects on users and groups and its associated roles, which is termed as *Phase* and *Lifecycle* here. Through the *Phase* and *Lifecycle* concept, applications are then able to specify the temporal aspect of users' information where sequencing is important. Phases are expected to be sequential in a lifecycle, and it is also possible to have multiple concurrent phases within a lifecycle, and these phases may even overlap other phases, as illustrated in Figure 2. This figure shows an example of the relationships between the *Phases and a Lifecycle* of a project.

The potential influence of this temporal aspects and the manner it may change the roles associated to users/groups are further illustrated through an example shown in Figure 3. In this example, a project lifecycle begins when the project is initiated and ends when the project is

completed or terminated. Along the timeline of the project's *Lifecycle*, there are sequential multiple *Phases*. In each phase, *Resources* (that could potentially be users) are needed to fulfil task(s) that are allocated to a particular *Phase*. The *Resources* could be added in the form of individual users, or in the form of groups of users, which means there are associated *Users* or *Groups* in each phase. Therefore, by associating the users (individual or groups) with a particular *Phase* in the project's *Lifecycle*, different *Role(s)* can then be assigned through individual *Phases,* thus fulfilling the requirement for *Roles* to be time sensitive.

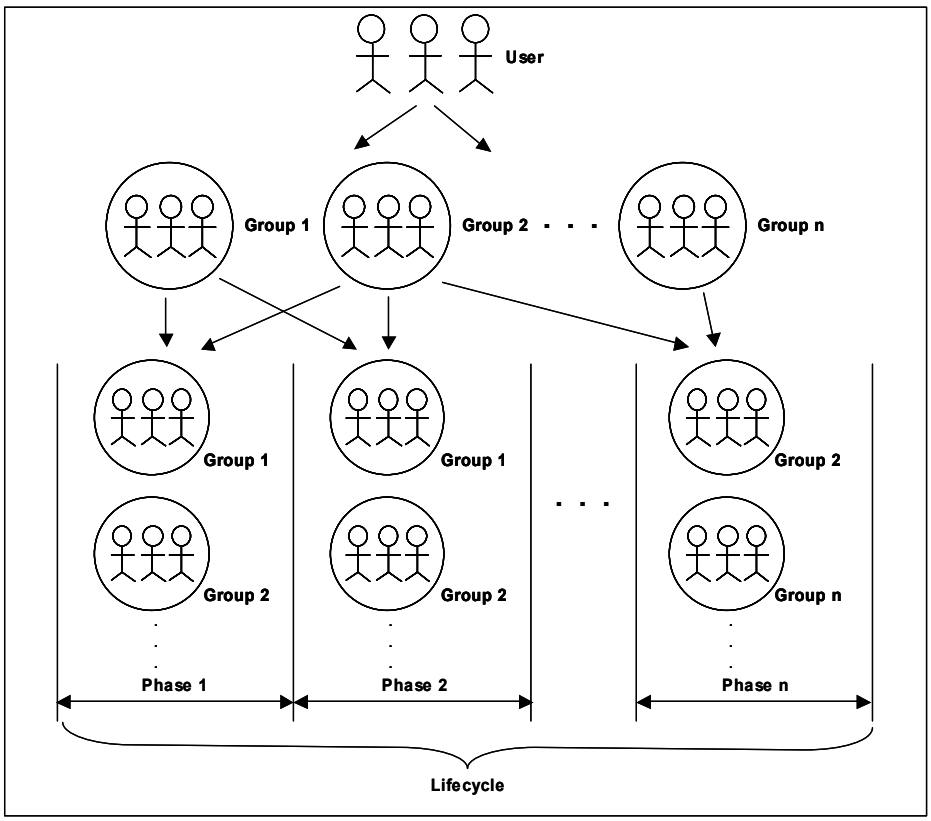

Figure 3: An illustration of Relationships between User/Group and Phase and Lifecycle

User is a form of *Resources*, another important form being services. The management of services are equally important, as like users, services need to be registered, updated and deleted for housekeeping purposes also. Furthermore, there is an extra need for services to be made known or discovered by potential service users or consumers. Like its counterpart, services also need to be tracked for its utilisation. Furthermore, as a service typically provides a specific set of functionality to its users/requestors, there is also a need to monitor its availability, performance and potentially its reliability. The ability to provide for sequencing of services or service aggregation is addressed in section 2.2 separately.

Data or information[3] is another form of Resources that needs to be managed. Management of data/information typically pertains to ensuring that correct data / information is transmitted and accessed and most importantly, the integrity of data / information is also ensured. For this, the security aspects are handled in section 2.4.

[3] Services and Data/information as other forms of important Resources in an application are expected to be detailed further in later versions.

2.1.1 Management of Resources

[MANAGEMENT-001] Provide authorised user(s) the ability to create, update and delete user accounts.

[MANAGEMENT-002] Facility to enable account owners to update or modify their own account information when needed.
Example of account information could include password, address, mobile phone number, Email address, etc.

[MANAGEMENT-003] Provide authorised user(s) the ability to view or query the user accounts including the account status based on pre-define selection criteria for monitoring purposes.

[MANAGEMENT-004] Provide authorised user(s) the ability to set personalised preferences.
Example of preferences could include colours, subscribed services, devices used, etc.

[MANAGEMENT-005] Facility to enable authorised user(s) to define, assign, update and delete user privileges whose accounts are under their charge or care.

[MANAGEMENT-006] Provide authorised user(s) the ability to log user activities.

[MANAGEMENT-007] Facility to enable different reports to be generated based on user activities through customisation or configuration.
Example of reports could be usage patterns and warning/error/fatal messages generated based on user account.

[MANAGEMENT-008] Facility to enable authorised user(s) to define at run-time the set of user information that needs to be captured.
Example of captured user information could be the extended set of default user information. The extended set could include user logs in time, the duration of the stay, the usage patterns, etc.

[MANAGEMENT-009] Facility to enable authorised user(s) to configure data sources for storing information.
Examples of data sources include text files, HTML, XML, LDAP, RDBMS and XML DB.

[MANAGEMENT-010] Provide authorised user(s) the ability to preserve user context information for reference.
Example of user context information include user last login time, services that user last access, user last noted access point, etc.

[MANAGEMENT-011] Provide authorised user(s) the ability to identify user's current location, time and service access patterns to provide personalisation services.

[MANAGEMENT-012] Provide authorised user(s) the ability to create, update and delete user definition.
Example of user definition could be the additional information required in addition to the basic set of user information collected when user accounts are created.

[MANAGEMENT-030] Provide authorised user(s) the ability to define roles for managing users.
Examples of roles could be user role in an organisation, user role in a project team, etc.
Therefore, roles could be defined for a user performing different tasks or job functions in various project groups, systems and organisations.

[MANAGEMENT-031] Provide authorised user(s) the ability to re-define role(s), update role(s) information and delete role(s) defined.

[MANAGEMENT-032] Provide authorised user(s) the ability to assign a role or multiple roles to a user.

[MANAGEMENT-033] Role or multiple roles could also be assigned to different groups of users in different phases of a project lifecycle.
For example, a user assigned a role of *Developer* could be assigned a different role as a *Tester* in a project Testing phase where he could be testing modules developed by other developers.

[MANAGEMENT-034] Provide authorised user(s) the ability to delete user roles when the roles become obsolete.

[MANAGEMENT-050] Provide authorised user(s) the ability to create new groups based on default group definition.
Examples of groups could be user groups, project groups, etc. Group definition could be group name, group description, etc.

[MANAGEMENT-051] Provide authorised user(s) the ability to dynamically define group definition.

[MANAGEMENT-052] Provide authorised user(s) the ability to create new groups based on the dynamically defined group definition.

[MANAGEMENT-053] Provide authorised user(s) the ability to retrieve groups, update group information and delete the group created.

[MANAGEMENT-070] Provide authorised user(s) the ability to define lifecycles.

[MANAGEMENT-071] Provide authorised user(s) the ability to re-define the lifecycles.

[MANAGEMENT-072] Provide authorised user(s) the ability to assign lifecycle to a users or group(s) of users.

[MANAGEMENT-073] Provide authorised user(s) the ability to retrieve and update lifecycle information and to perform housekeeping.
Example of housekeeping could be to delete the lifecycle information when it is obsolete.

[MANAGEMENT-074] Provide authorised user(s) the ability to define phases.
Example for phases could be project development phases such as Architectural Design and Implementation Phases.

[MANAGEMENT-075] Provide authorised user(s) the ability to re-define the phase.

[MANAGEMENT-076] Provide authorised user(s) the ability to assign phases to lifecycle.

[MANAGEMENT-077] Provide authorised user(s) the ability to retrieve, update and delete phase information.

[MANAGEMENT-078] Provide the ability for FEs to manage resources for multiple applications.

[MANAGEMENT-090] Facility to enable automatic discovery of services within specified domain(s) for monitoring purposes. If the service is an aggregation, then the dependent services should also be discovered [4].

[MANAGEMENT-091] Facility to enable automatic generation of service client(s) from a WSDL. With this service client, it would be possible to invoke the services and monitor the invocations, specifically the turn around time of invoking the Web Service [4].

[MANAGEMENT-092] Facility to provide the service client(s) with the appropriate test data.
Example of test data could be a XML file that conform to a certain XML schema definition [4].

[MANAGEMENT-093] Facility to enable authorised user(s) to monitor server resources, operation system resources, usage and performance of the hosting server [4].

[MANAGEMENT-094] Facility to enable authorised user(s) to check or monitor service(s) availability [3].

[MANAGEMENT-095] Facility to enable authorised user(s) to log or track the service usage pattern and also monitor the service status [4].

[MANAGEMENT-096] Provide authorised user(s) the ability to maintain additional service attributes such as access privileges and support for output devices.

[MANAGEMENT-097] Provide authorised user(s) the ability to create, update and delete service categories.

[MANAGEMENT-098] Provide authorised user(s) the ability to register, update and delete services and their related information under a service category.

[MANAGEMENT-099] Provide the ability to search the service categories and the services registered under each service categories.

[MANAGEMENT-100] Provide the ability to discover published Web Services.

[MANAGEMENT-110] Facility to enable authorised user(s) to define, retrieve and delete log categories.
Example of log category could be the data sources (for example, XML files), the data fields in each data source that user(s) want to log.

[MANAGEMENT-111] Facility to enable authorised service(s) to log events.

[MANAGEMENT-112] Provide authorised user(s) the ability to open the log file for further processing of the log information such as performing analysis from the log information.

[MANAGEMENT-113] Provide authorised user(s) the ability to search through any log records based on conditions.
Example of conditions could be field names.

[MANAGEMENT-114] Provide authorised user(s) the ability to backup or archive log records and then delete them when they are not needed.

2.1.2 Management of Access to Resources

[MANAGEMENT-200] Provide authorised user(s) the ability to define access structure for managing users, information and services.

[MANAGEMENT-201] Provide authorised user(s) the ability to re-define the access structure or update the access structure information.

[MANAGEMENT-202] Provide authorised user(s) the ability to assign different access levels to roles based on the defined access structure.

[MANAGEMENT-203] Provide authorised user(s) the ability to update and unassigned access level assigned to roles.

[MANAGEMENT-204] Provide authorised user(s) the ability to give access permissions to resources based on the role the user been assigned to.

[MANAGEMENT-205] Provide authorised user(s) the ability to receive alert when unauthorised access is detected by system.
Alert could be received through email, SMS, etc.

2.2 Process

Typically, applications are built to satisfy a set of business needs. These business needs are often represented through a series of business processes. A business process enacts a sequence of activities, the flow of data among the activities and the manipulation of events that occurs in a value chain, which could be within a specific platform, across different platforms within an enterprise or even across enterprises. The successful composition and execution of these activities or tasks are an integral part to any application, and especially more so to Web Service enabled-applications that operate in a distributed environment of the SOA model.

In this section, the *Process* category [3] handles aspects that are related to enabling the composition and execution of a sequence of tasks, which include handling requirements arising out of executing these tasks in a sequenced manner, in a distributed environment that typically comprises more than one interacting services. As such, this category has two broad aspects to be considered, namely *Workflow* and *Invocation.*

Under *Workflow*, requirements arising out of composition, aggregation and sequencing of planned tasks of a business process are considered. In a SOA model, functionalities that help fulfil a business process typically reside across services that could operate on different applications across different platforms. The ability for a Web Service enabled-application to harness these different services into an aggregated business process as a more value-added service is critical and non-trivial. Here, the composition, aggregation and potentially orchestration activities could be articulated based on standards like BPEL or WSCI for example or could even be hard-coded implementations.

Under *Invocation*, the requirements arising out of the actual execution of planned tasks are handled. This handles aspects arising from the actual execution of services within a defined business process, including ensuring that services for fulfilling the tasks are available for execution, and the performance or reliability of these said services are within acceptable ranges for example. Furthermore, in a business process, activities or tasks typically trigger actions that must be taken to fulfil the task(s) or activation of other tasks within that business process. Monitoring and handling of such actions or events constitute an integral part of business process management. There is a strong demand for improving the time lag between each business activity, requiring more timely responses, which in turn leads to the need for faster and more automatic handling of actions/events arising of activities within business processes. This is harnessed into an Event Handling concept. It enables real time monitoring, notification, response of events and collaboration of handling of events across people, disparate data sources and applications, which can be both internal within a company and external to the company.

2.2.1 Workflow

[PROCESS-001] Provide the ability to ease the tasks of administering and configuring processes. The tasks should include but not limited to the following:

- Create an individual process instance.
- Start a process instance.
- Terminate a process instance.
- Assign a priority for a process.
- Assign access rules to control a process.

[PROCESS-002] Provide the ability to specify workflow process logic, process participants and rules governing participation without the need to do programming.

[PROCESS-003] Provide the ability to analyse the process structure during the process design phase by having the following but not limited to:

- Validate the process to ensure its correct behaviour.
- Verify the process for redundancy of activities.
- Test correct data accessed by various process.

[PROCESS-004] Provide the ability to analyse and monitor the execution of individual processes.

[PROCESS-005] Provide the ability to send warning messages when any process deadline is missed or activity is not activated.

[PROCESS-006] Provide the ability to manage the behaviour of the business processes such that analysis could be done to work on the optimisation of that process when required.

[PROCESS-007] Provide the ability to monitor the status of individual process such as the following but not limited to:

- Process initiation.
- Live processes (running).
- Process suspension.
- Process completion.
- Process termination.

[PROCESS-008] Provide authorised user(s) the ability to record messages during the execution of a process.
Examples of messages to be recorded include warning messages, error messages, fatal error messages, etc.

[PROCESS-009] Provide authorised user(s) the ability to view all records or logs.

[PROCESS-010] Provide the ability to define execution rule(s) for service composition based on a defined process.
Examples of the execution rule(s) should include the specification of relationships, conditions and parameters of service execution, etc.

[PROCESS-011] Provide the ability to manage the execution rule(s) to facilitate the addition, update and deletion of defined rules.

[PROCESS-012] Provide the ability to aggregate new services from existing services based on the defined execution rules.
For example, if there is an airline ticketing service, a hotel reservation service and a car rental booking service, there is possibility that these 3 can work together, without human intervention, to provide an aggregated service based on the defined execution rules.

[PROCESS-013] Provide the ability to modify aggregated service(s) by adding, deleting and changing the Web Services.

[PROCESS-014] Provide the ability to publish newly aggregated services to specified Registry server(s) that are UDDI or ebXML Registry compliant.

[PROCESS-030] Facility to enable authorised user(s) to administer and configure multiple data sources.

[PROCESS-031] Provide the ability to locate and discover existing data sources.

[PROCESS-032] Provide the ability to locate and discover existing Web Services.

[PROCESS-033] Provide the ability to store the content provided by the content provider in cache and provide optimal time interval for updating the cache.

[PROCESS-034] Provide the ability to be flexible and adaptable to new types of data sources.

[PROCESS-035] Provide the ability to be flexible and adaptable to new data types of existing data sources.

2.2.2 Invocation

[PROCESS-100] Provide the ability to create, retrieve, update and delete event types dynamically.

[PROCESS-101] Provide the ability to define dynamically the processing logic for pre-defined event types.
The processing logic could consist of single or multiple tasks to route the event to the channels or to the consumers who subscribed to the event.

[PROCESS-102] Provide subscriber(s) the ability to receive notification(s) when the event he subscribes occurs.
Examples of notification(s) received could be through Email, SMS, etc.

[PROCESS-103] Provide the ability to detect when the pre-defined event occurs in external systems.
Examples of external systems could be legacy system, inventory management system, etc.

[PROCESS-104] Provide the ability to notify an event based on the pre-defined processing rule.
The pre-defined processing rule could include the condition, time and event types to be triggered.

[PROCESS-105] Provide the ability to schedule the activation of invocation(s) and notification(s) to occur at specific time when triggered by an event.

[PROCESS-106] Provide the ability to create, retrieve, update and delete event supplier(s) and its related information.
Examples of related information could be name, description, registered date, etc.

[PROCESS-107] Provide the event supplier(s) the ability to notify the event.
Examples of notification sent could be by Email, SMS or other ways of Web Service invocation.

[PROCESS-108] Provide the ability to create, retrieve, update or delete the event consumer(s) and its related information.
Examples of related information could be name, description, registered date, Email address, mobile phone number, etc.

[PROCESS-109] Provide event consumer(s) the ability to subscribe to an event or channel to be notified when the event occurs or when the subscribed channel receives the event.

[PROCESS-110] Provide event consumer(s) the ability to receive notification(s) when the subscribed event occurs.
Examples of notification(s) received could be through Email, SMS, etc.

[PROCESS-111] Provide the ability to create, retrieve, update and delete a channel and its related information.
Examples of related information could be name, description, created date, etc.

[PROCESS-112] Provide authorised user(s) the ability to define filters.

[PROCESS-113] Provide authorised user(s) the ability to attach filter to either an event or a channel.

[PROCESS-114] Provide the ability to retrieve, update or delete a defined filter.

[PROCESS-115] Provide the ability to log the occurrence of events.

[PROCESS-116] Provide the ability to log notification of events to event consumers.

[PROCESS-117] Provide the ability to support event notification in synchronous and asynchronous SOAP messages (request and response) based on pre-defined message format.

[PROCESS-118] Provide the ability to schedule messages to be sent at designated time(s). Example of devices could be Email, SMS, etc.

[PROCESS-130] Provide the ability to test the services deployed.

[PROCESS-131] Provide the ability to test the aggregated services deployed.

[PROCESS-132] Provide the ability to test the availability of services.

[PROCESS-133] Provide the ability to test the availability of the delivery channels. Example of delivery channels could be Web Server, Application Server, SMS/WAP server, etc.

2.3 Delivery

The last few years have seen a dramatic rise in the types of devices and access mechanisms available to end users when accessing or retrieving information from the virtual world. Devices range from personal computers to appliances in the home to mobile phones and PDAs. Access mechanisms have also proliferated, from traditional wired access to wireless access with different bandwidth capabilities through these different devices. End users now expect to be able to access critical information through different access mechanisms from different locations at any time of the day. As such, applications are now expected to deliver information to these end users in a variety of ways, according to the required display format, access mechanism and bandwidth capabilities of the device in use.

For Web Service-enabled applications, there is also a need to manage the way SOAP information is transformed and understood by other services or applications. Typically this could include:

1. *Mapping of one service output(s) as input(s) to another service in a SOAP form*[4]
   This is typically needed in an aggregated service where one service output is expected to form the whole or part of the input to another service.
2. *Transformation of the SOAP message into an acceptable native form* of an application and vice-versa[5]
   This aspect is usually required when Web Services interact with legacy applications that are typically implemented in a non-XML or Web Service way.
3. *Transformation of the SOAP message into a preferred display format* to cater for different device needs and vice-versa
   This is usually needed at the interface layer between an application and its users, and is expected to be delivered in an interactive manner, as opposed to the automatic manner of the first two forms.

The requirements listed here cover aspects of conveying, producing or transferring content to desired destinations or devices in the preferred format using selected delivery mechanisms [3], and are skewed towards the 3rd form discussed above.

[DELIVERY-001] Provide the ability to deliver the data or content to devices in the preferred format.
Examples of devices could be browser, handheld devices, etc.

[DELIVERY-002] Provide the ability to customise or state the delivery and rendering preferences.

[DELIVERY-003] Provide the ability to send the status of delivery.
Examples could be the status of message queue by notification, acknowledgement, etc.

[DELIVERY-004] Provide the ability to track the status or the location of delivery by searching and logging the state of the delivery.

[DELIVERY-005] Provide the ability to configure delivery format depending on devices.
Examples of delivery formats could be MIME type, kind of display, etc.

[DELIVERY-006] Provide the ability to convert SOAP to the preferred output type.
Examples of output type could be in the format of HTML, cHTML, WML, etc.

[DELIVERY-007] Provide the ability to format the output into reports or graphs.

[DELIVERY-008] Enable the ability to provide language agnostic delivery.

---

[4] This aspect is presently not covered in this draft. Future drafts or versions may include this aspect for consideration in the Functional Elements Specification.

[5] This is expected to be detailed in later versions.

Examples of languages could be English, Chinese or Malay that is preferred by the receivers.

[DELIVERY-009] Provide the ability to have location-based delivery.

[DELIVERY-010] Provide the ability to intercept request and check for the supported MIME type(s) in application to return an appropriate result.

2.4 Security

Security is a key aspect in all applications, more so for a Web Service-enabled application that is externally accessible. This is due to the fact that a Web Service-enabled application will typically harness resources from different services across enterprises, and such communications and data exchange are prone to malicious tampering and attacks if not protected adequately. Furthermore, in a SOA environment, controlled and authenticated access to services is also crucial, to ensure that malicious or unauthorised access to services do not take place.

Under the topic of security the following five areas are usually discussed, namely:

- *Authentication*
  This relates to the security aspect that handles the positive verification of the identity of a user, device, or other entity in a computer system. It is often as a prerequisite for access to *Resources* in a system.
- *Authorisation*
  This relates to the process of determining whether a subject is allowed specified types of access to a particular *Resource*. This is typically done by evaluating applicable access control information. Usually, authorisation is in the context of authentication. Once a subject is authenticated, it may be authorised to perform different types of access requested.
- *Confidentiality*
  This relates to assuring information is kept secret or confidential, with access limited to authorised users only.
- *Data Integrity*
  This relates to assuring information will not be accidentally or maliciously altered or destroyed.
- *Non-repudiation*
  This relates to method(s) by which the sender of data is provided proof of delivery and the recipient is assured of the sender's identity, so that neither party can deny that a transaction or data has been completed.

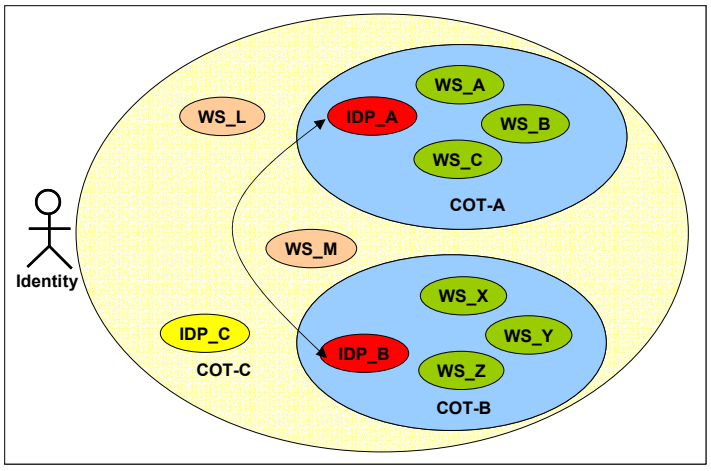

Figure 4: Circle of Trusts (COTs) within a Domain and Across Domains

In a Web Service-enabled application, it is imperative that security covers the following areas to ensure that the system and transactions are not compromised, and the five areas articulated are fulfilled:

- *Protecting User's Identity*
  This can be in the form of providing a means to specify the identity of the user and a mechanism to authenticate this identity. In a Web Service-enabled application, providing means to specify a user identity across different services or applications is important as the application is expected to aggregate functionalities from such services. In a SOA environment, the concept of Circle of Trust (COT) where a user (once the identity has been proven) is able to access services in the circle based on the initial authentication is crucial, be it based on Single-Sign-On within a domain or federation of identities across domains. Figure 4 illustrates this concept where COT-A and COT-B are formed within the same domain, i.e. SSO whereas COT-C are formed across two domains, i.e. using a Federated Identity concept. Furthermore, in each COT, at least one Identity Provider (IDP) is expected. In this area, the authentication and authorisation (rights/access assignments) aspects are important.
- *Securing Data Exchange*
  This can be provided in two ways, namely securing the message contents and the message delivery channel. In both cases, the protection can be in the form of encrypting the message or delivery channel to ensure that the confidentiality of the message is achieved. In addition, the message be digitally signed using a digital signature, to guarantee that the message's integrity and authenticity is preserved. As such, the data integrity and confidentiality are key aspects in this area.
- *Safeguarding of Resources* (including Web Services)
  This is important to prevent unauthorised access and usage of *Resources* like Web Services, and potential damage to the business data and assets. For example, a mechanism can be provided to Web Services for limiting access to privileged users only. Also, all Web Services accesses and transactions have to be diligently logged and acknowledged to prevent both legitimate and illegitimate users from denying having performed the transaction, which is the non-repudiation aspect. The access and control mechanisms for *Resources* are detailed in section 2.1

2.4.1 Protecting User's Identity

[SECURITY-001] Provide a facility to store, access and update user identity information using an identity repository.
Examples of identity repositories can be databases, files, etc.

[SECURITY-002] Enable the ability to authenticate users, at the very minimum, using username and password. Other authentication means can include smart card, biometrics, etc.
In the case where the authentication method is through username and password, the following additional requirements should be considered:

[SECURITY-002-1] Provide the ability to set password format.
[SECURITY-002-2] Provide the ability to generate new password.
[SECURITY-002-3] Provide the ability to notify user before the password expires.
[SECURITY-002-4] Facility to enable authorised user(s) to choose a suitable algorithm to encrypt password.

[SECURITY-003] Provide authorised user(s) the ability to access several applications or services like Web Services within a trusted domain without re-authentication after authentication has been done once.

[SECURITY-003-1] Provide a facility to request for authentication assertions from an Assertion Authority.
Example of Assertion Authority is SAML Authentication Assertion Authority.
[SECURITY-003-2] Provide the ability to do Single Sign-On (SSO) based on SAML assertions.

[SECURITY-003-3] Provide the ability to encrypt and decrypt sensitive information based on XML Encryption.

[SECURITY-004] Provide authorised user(s) the ability to access several applications or services like Web Services within a federated trust circle without re-authentication after authentication has been done once.

[SECURITY-004-1] Provide a facility to manage and federate multiple identities within a circle of trust using an Identity Provider.

[SECURITY-004-2] Provide the ability to do Federated Single Sign-On (SSO) based on an established standard.
Example of standard is Liberty Alliance or WS-Federation.

2.4.2 Securing Data Exchange

[SECURITY-020] Provide the ability to enable message security for SOAP messages such as XML signature and encryption. In addition, the following requirements listed below should be considered:

[SECURITY-020-1] Provide the ability to have XML digital signature based on XML Digital Signature specification.

[SECURITY-020-2] Provide the ability to canonicalise XML with/without comments.

[SECURITY-020-3] Provide the ability to perform a message digest on the data.

[SECURITY-020-4] Provide the ability to validate signed information.

[SECURITY-020-5] Provide the ability to encrypt either bulk information or sensitive part of information with symmetric/asymmetric key.

[SECURITY-020-6] Provide the ability to decrypt the encrypted information.

[SECURITY-021] Provide a mechanism to secure the delivery channel such as SSL/TLS.

[SECURITY-022] Provide the ability to manage private and public keys.

[SECURITY-023] Provide the ability to build key agreement between two parties.

[SECURITY-024] Provide the ability to import trust certificate.

[SECURITY-025] Provide the ability to revoke key based on Public Key Infrastructure (PKI).

[SECURITY-026] Provide the ability to authenticate the digital certificate(s).

2.4.3 Safeguarding of Resources

[SECURITY-040] Facility to enable definition of policy for specific user's role to protected resources.

[SECURITY-041] Facility to verify authorised user(s) based on the defined policies.

[SECURITY-042] Facility to define an audit profile for resources.
Example of audit profile could be usage activities to log and track access to resources.

[SECURITY-043] Facility to start and stop the execution of the audit based on the audit profile defined.

[SECURITY-044] Facility to analyse and manage audited information that has been logged.

[SECURITY-045] Provide the ability to create, retrieve, update and delete a defined audit profile.

[SECURITY-046] Provide the ability to backup the audit information.

[SECURITY-047] Provide the ability to delete the audit information.

3 References

1. A reference of the Web Services Implementation Methodology will be provided at a later date.

2. **Web Services and Service-Oriented Architectures**, May 2004, http://www.service-architecture.com/web-services/articles/service-oriented_architecture_soa_definition.html

3. Ang C.H., Tan P.S., et. al., **Reference Architecture Requirements Specifications**, version 1.1 of Release 1.0, July 2003 by Singapore Institute of Manufacturing Technology.

4. Cheng. J., Cheng, Y.S., **Service Management Requirement Specifications**, version 0.1 of Release 1.0, February 2003 by Singapore Institute of Manufacturing Technology.

Appendix A. Acknowledgments

The following individuals were members of the committee during the development of this specification:

- Christopher Haddad, Individual
- Eng Wah Lee, Singapore Institute of Manufacturing Technology
- V. Ramasamy, Singapore Institute of Manufacturing Technology
- Zun Liang Yin, Singapore Institute of Manufacturing Technology
- Lim Kenneth, CrimsonLogic Pte Ltd
- Ravi Shankar, CrimsonLogic Pte Ltd
- Jagdip Talla, CrimsonLogic Pte Ltd
- Andy Tan, Individual
- Roberto Pascual, Infocomm Development Authority (IDA) of Singapore

The committee would also like to express its appreciation for the encouragement and guidance provided by Jamie Clark throughout the course of the TC work.

Appendix B. Revision History

Rev	Date	By Whom	What
FWSI-FESC-Requirements-01.doc	31-May-04	Tan Puay Siew Ang Chai Hong	First Draft
FWSI-FESC-Requirements-01a.doc	01-Jul-04	Ang Chai Hong Tan Puay Siew	Minor changes to Individual Requirements of First Draft - • **MANAGEMENT** - Added 012, 078, 097, 098, 099 and 100 • In **[MANAGEMENT-090]**, added "within specified domain(s) for monitoring purposes • **PROCESS** – Added 035, 117 and 118 • In **[PROCESS-009]**, added logs • In **[PROCESS-031]**, removed "or services" • Changed **[PROCESS-032]** requirements • In **[PROCESS-034]**, removed the word "data" • In **[PROCESS-104]** replaced the word "trigger" with "notify" • **DELIVERY** – Added 010 • In **[DELIVERY-001]**, added "in the preferred format" after devices • In **[DELIVERY-003]**, replaced the word receive with send • **SECURITY** – Added 045, 046 and 047 • Reworked **[SECURITY-022]** into 5 requirements ranging from **[SECURITY-022]** to **[SECURITY-026]** • In **[SECURITY-043]**, added the word "stop"

Appendix C. `Notices

OASIS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on OASIS's procedures with respect to rights in OASIS specifications can be found at the OASIS website. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementors or users of this specification, can be obtained from the OASIS Executive Director.

OASIS invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to implement this specification. Please address the information to the OASIS Executive Director.