OASIS Business-Centric Methodology Appendix B: Linking and Switching

OASIS Standard
2006 May 1

Document identifier: BCM.AppendixB.LinkingSwitching.2006-05-01.doc

Location: http://www.oasis-open.org/committees/bcm/docs/

Send comments to: bcm-comment@lists.oasis-open.org

Editor:

David RR Webber, (drrwebber@acm.org)

Contributors:

Bruce Peat, eProcess Solutions (emeritus Co-Chair)

Mike Lubash, Defense Finance and Accounting Service -DFAS (Co-Chair)

Carl Mattocks, Checkmi (Co-Chair)

David RR Webber, consultant (Secretary)

Eric Okin, Defense Finance and Accounting Service -DFAS

Hans Aanesen, Individual

Sally St. Amand, Individual

Laila Moretto, MITRE

Dan Pattyn, Individual

Paul Kirk, Hewlett-Packard

Neil Wasserman, Individual

Bob Greeves, US Department of Justice EOUSA

Michael Callahan, Attachmate

Murali Iyengar, SAIC

Tony Fletcher, Choreology Ltd

Zachary Alexander, Individual

Sergey Lukyanchikov, Individual

Anders Tell, Individual

Riaz Kapadia, Infosys Technologies

Vishwanath Shenoy, Infosys Technologies

Moshe Silverstein, iWay Software

Marc Le Maitre, OneName Corporation

Kumar Sivaraman, SeeBeyond Technology Corporation

Puay Siew Tan, Singapore Institute of Manufacturing Technology

Arne Berre, SINTEF

Paul Boos, US Dept of the Navy

Abstract:

This specification appendix defines the BCM V1 Linking and Switching details.

Status:

This is an OASIS Standard document produced by the Business-Centric Methodology Committee. It was approved by the OASIS membership on 19 April 2006.

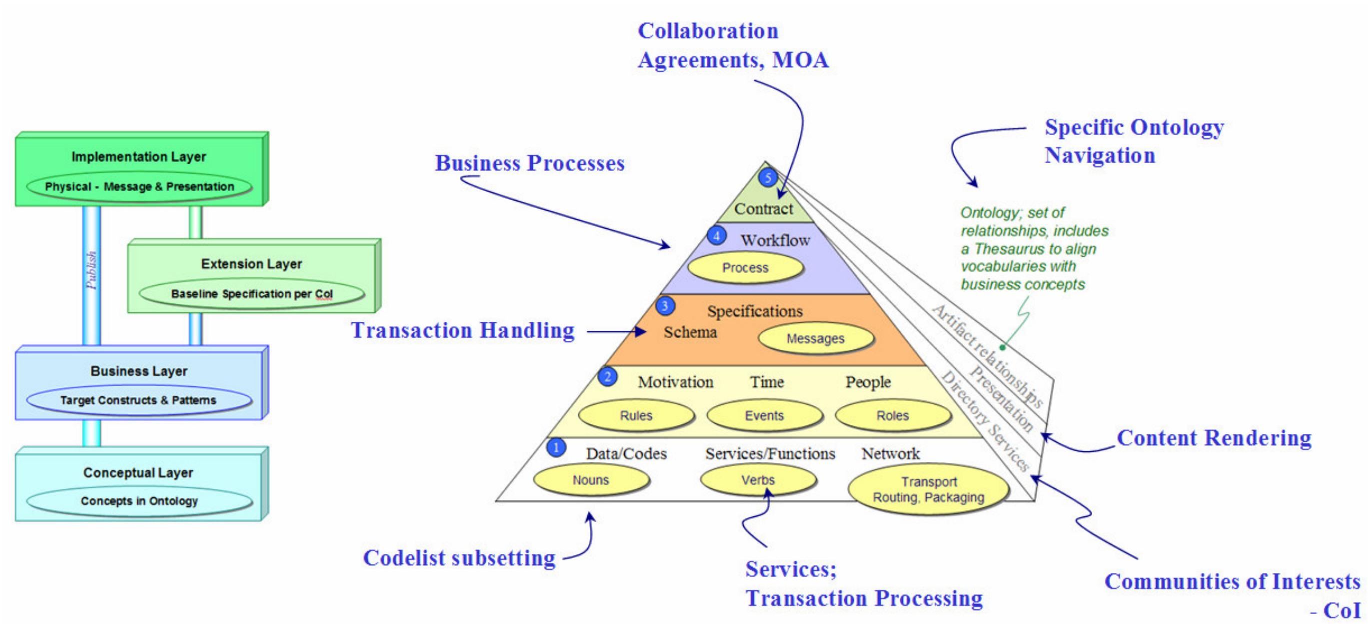
If you are on the BCM list for committee members, send comments there. If you are not on that list, subscribe to the bcm-comment@lists.oasis-open.org list and send comments there. To subscribe, send an email message to bcm-comment-request@lists.oasis-open.org with the word "subscribe" as the body of the message.

Table of contents

1. The Linking and Switching Environment

The focus of the BCM approach is in providing the understanding to allow enterprises to acquire and sustain agile information systems that provide reliable business exchanges between stakeholders. In analyzing prior legacy approaches and in place systems one key factor is the inability to support context driven processes and information exchanges dynamically. Particularly in place systems where the logic control is hardwired into program code or locked into proprietary delivery systems are inhibitors to agile information exchanges themselves and any mitigation or migration techniques seeking to bypass those restrictions. Figure 1 depicts some of the context-based switching that occurs at each of the BCM layers within the information architecture, along with those which occur at Conceptual, Business, Extension, and Implementation layers.

Figure 1 – Need for Context-based Linking and Switching



Today with the advent of individual implementation technologies including XML driven software mechanisms, open standards for e-Business transaction formats, and web service aware components the challenge is in configuring these to support dynamic context, semantics and syntax for interoperable business exchanges. Ironically these same challenges have already been architected and tackled previously by agent driven systems designed for dynamic decision support. However those prior agent systems suffered from using proprietary interfaces and rule bases so that they could not interoperate easily. Instead by using open shared concepts that are business-centric and linked to XML formats and exchange mechanisms this shortcoming can be addressed (some work has already been done in this direction with efforts such as RuleML and BRML[1], however these have not focused specifically on the business needs and supporting those mechanisms directly).

The next challenge is ensuring that deployed components actually support the open specifications mechanisms in a consistent way. Then it becomes possible to create the agile information exchange systems that users can exploit using a “business-first through choice” doctrine. This is the focus of the BCM approach, and this section of the BCM specification details how *Choice Point* mechanisms are needed to enable context driven agile information exchanges that allow the use of linking and switching across the individual components.

Choice Points can be seen as providing three enablers for agile information exchanges:

- Context criteria, where the scope of the context extends beyond the local decision point, and can also require persistence of decisions

- Determining context by refining criteria dynamically, and that may include undetermined start points

- Where the context requires a thread manager to establish and track the state of a process.

There are other significant aspects to the implementation of Choice Points, such as consistent semantic definitions for the context rules and robust process control syntax that allow the user business requirements to be precisely defined. Those aspects are discussed elsewhere in the BCM specifications and merely noted where applicable in this section. Also the use of the Choice Point approach does significantly enhance these other areas, since it is a broad horizontally applicable technique that can be used to manage all aspects of agile information exchanges. This serves to highlight the difference with today’s systems that lack Choice Point technology. Such non-agile systems are therefore static inflexible ‘stovepipe’ solutions that cannot support dynamic linking and switching and are thus hard to re-purpose and change.

A further significant benefit of the Choice Point approach is that it exposes and makes available the context parameters within a given application layer. This allows business decisions and choices to be clearly known, classified and selected. Whereas previously applications were built as a “black box” that could not be easily re-purposed or their suitability to task quickly determined.

Next we consider the implementation constraints. The intention here is to provide a neutral definition of the BCM Choice Point mechanisms and their XML representations that implementers can then construct and integrate using popular rule engines. Since each application own needs will vary it is important that implementers can choose to build just a tailored sub-set while maintaining interoperability across Choice Points as a prime requirement. This includes the ability to scale

[1] RuleML – Rules Markup Language and BRML - Business Rules Markup Language and others – complete list is available with links at: http://xml.coverpages.org/ruleML.html

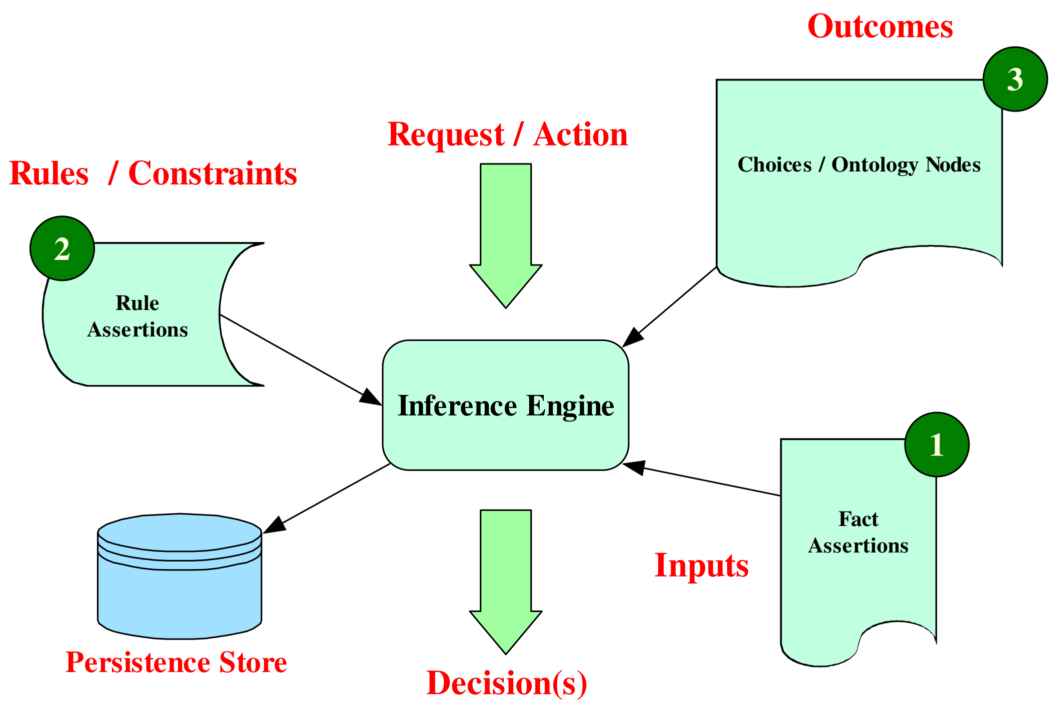
linearly from a simple Choice Point with a single rule-set through to a decision support rule engine operating on a dynamic knowledge base with thousands of facts and rules.

In order to implement Choice Point technology requires the ability to manage the inputs (facts) and outputs (choices) and rule mechanisms applicable to a choice using open consistent formats in XML and communication protocol standards (see the Choice Point template diagram in figure 2 below). These mechanisms should be “business-first” and accessible to business user audiences and technical business analysts. This paper details the steps needed in developing this approach and how that aligns with the overall main body of the BCM specifications.

2. Choice Points – Declarative Context-based Switching

The BCM approach emphasizes the need to understand the business problem domain and then translate that by layers into physical implementation logic and semantic constructs. Part of that process is defining Choice Points within the layers providing the means to capture and implement the decision logic. In addition understanding the ontology associated with those Choice Points is also required.

Figure 2 – Choice Point Conceptual Overview



As noted in the previous section the Choice Point consists of inputs, business rules and outputs that determine the linking and switching to be provided within the business exchange(s). In order to configure a Choice Point the business functional needs must be considered and detailed.

Within the BCM layers[2] there is the need to identify various key interactions and primitive entities that describe an interoperable business scenario. These include partner definitions, collaborations and roles, process definitions, information transactions and semantic details. Using this set of factors and participants we can then state the following:

---

[2] The diagrams of the BCM layers can be downloaded as large posters from http://dfas.info

- Qualifying context is key to ensuring correct relationships between partners in business collaborations
- Knowing context is needed to ensure accurate information capture, packaging and delivery
- Lack of context control (of the processing and transactions) is the single most prominent reason why legacy e-Business systems are complex to implement and support
- Providing and managing context is needed to drive dynamic process configuring and control
- Defining ontology both of the Choice Points themselves and including Choice Points within ontologies (see figure 3 below).

The context mechanism itself needs to be multifaceted in the types of decision choices that can be determined and controlled.

Context can be viewed as a series of cascading Choice Points that have inputs through the assertion of facts, the operation of rules and constraints, which determine the outcomes from available choices. These range from the very simple – "if then do" style - to event handlers, to state management, to complex decision agents that operate on sets of dynamic facts that include status information about concurrent operations.

Of course implementations must be able to choose how simple or complex their needs are and implement Choice Points accordingly. The rules selections may vary from simple binary choices through to complex decision support questions such as "buy or repair?" logistics. The BCM Choice Point approach is designed to scale from the simple to the complex in a linear and consistent way.

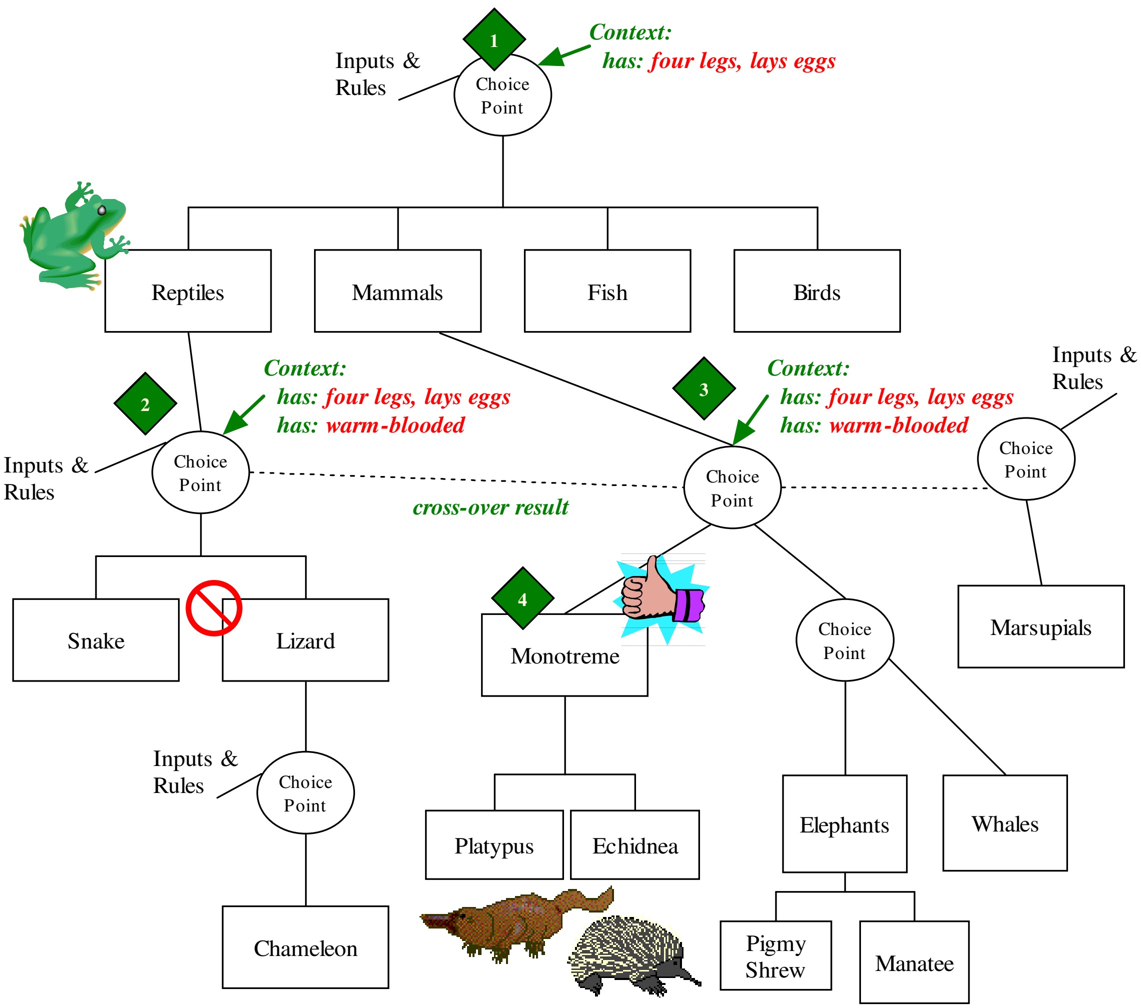
The Choice Point approach lends itself to today's *web service* technology. A Choice Point can function as a web service, or set of web service calls, that provide dynamic control and decision-making. Or the Choice Point can be a local component that references assertions and facts from a web service. Typical uses include tracking and controlling business processes, building transaction content and providing status of discreet events.

In examining context to determine the needs it is important to identify that context comes in many flavours and we can detail the more important types in order that these can be quantified for a particular implementation. Notice also that context flows through the four layers from the BCM architecture of conceptual, business, extension and implementation layers.

Typically the first context that is needed is to determine the Community of Interest (CoI). This enables one to then exploit re-use by searching within that CoI for components that may be adapted for the current purpose.

Next are the business agreement context and the business agreement roles that equate to the business purpose. Once these are established then the classification of artifacts within that context can be determined. Classification is a powerful tool for rapidly locating related context and determining which selection is appropriate from those available. Therefore a classification hierarchy may contain implicit context switches, or actual Choice Point components (see figure 3 for an example of a contextual hierarchy) that can be traversed, and the branching that may occur across the hierarchy based on relations and associations[3].

Figure 3 – contextual classification hierarchy with crossovers



Continuing with the analysis of context types into the implementation layer from which understanding the business process is paramount. This includes process selection context and process tracking context. Below the process is the transaction context followed by the exception-handling context. At the interface to the application systems there is context that is supplied to the decisions and rules that are associated with the information handling.

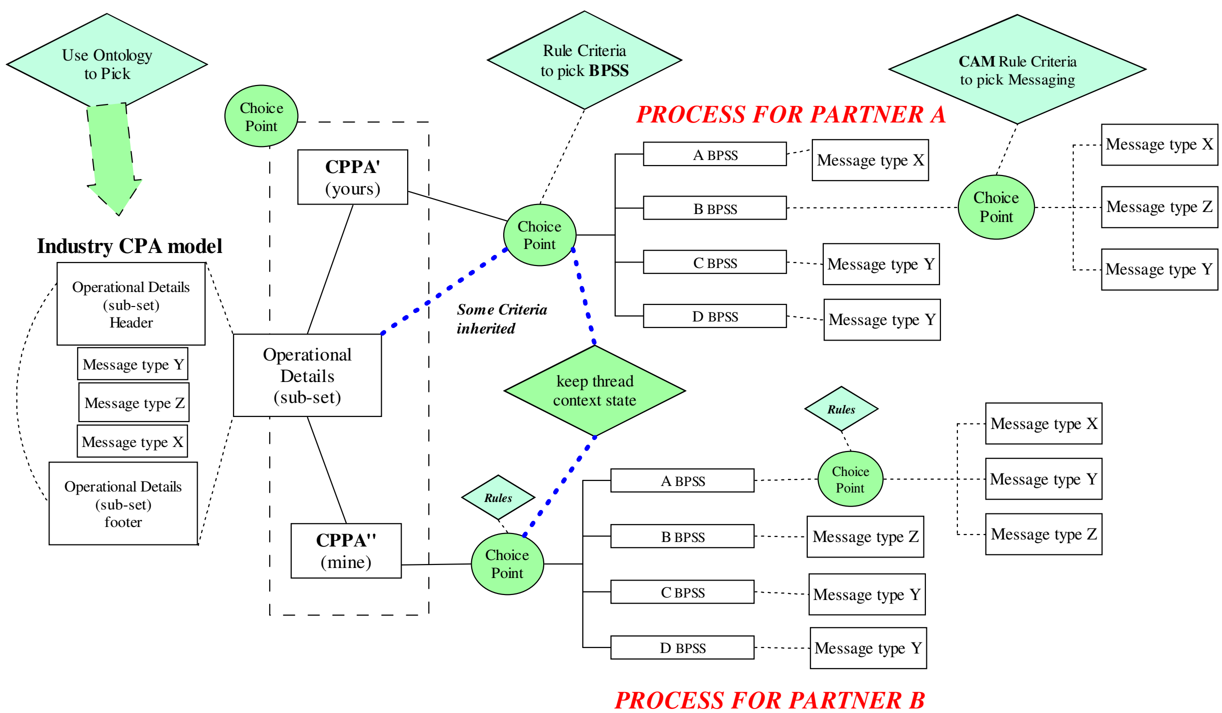
---

[3] Note: ebXML registry information model fully supports this use and the 'browse and drilldown' approach.

This cascading of Choice Points through the business implementation layers can be seen in figure 4, where the ebXML implementation stack[4] is used as an example. The context can be summarized as the following:

- Community of interest determination (CPPA specification / business ontology)
- Business agreement context (CPPA specification)
- Business agreement roles (CPPA specification)
- Classification of artifacts context (CPPA specification)
- Process selection context (BPSS specification)
- Process tracking context (BPSS specification)
- Transaction context (BPSS specification / CAM specification)
- Exception handling context (CAM specification)
- Decisions and rules context (CAM specification)
- Lookup tables and contextual subsets (CAM specification)

Figure 4 – Cascading e-Business choice points within the implementation layer



---

[4] CPPA – Collaboration Partner Profile Agreement (ebXML), BPSS – Business Process Schema Specification (ebXML), CAM – Content Assembly Mechanism (OASIS).

Reviewing figure 4 from left to right, the initial step is to use the ontology to determine the correct community of interest and select the model for the business exchange required. The model will include details of the business process and the document exchanges (as shown with the header and footer. Each trading partner then refines these based on their own operational details, and creates a Choice Point set of inputs, rules and outcomes based on the model. They then compare these and agree on the specific business process(es) they wish to use, the transaction messages (their structure format, content semantics and content rules), and update the context criteria accordingly to enforce these. These actions correspond to determining the context items summarized in the list immediately above figure 4.

The thread context state mechanism shown linked between the Choice Points allows both partners to keep in lock step with each other's business processes as the actual exchanges occur in their real-world systems (thread management is part of the Choice Point functional requirements already noted earlier).

Figure 4 shows a wide variety of possible business process paths and message choices with four process sequences (A,B,C,D) and three message formats (X, Y, Z). Typically business partners would pick just a subset of these for their initial implementation needs.

Choice Points therefore are involved in the entire process; configuring the business partner collaborations, selecting the details of the business processing, controlling the transaction content messages and tracking the state of each interchange that occurs.

As previously noted the Choice Point approach lends itself to today's web service technology as part of a Service Oriented Architecture (SOA). Each Choice Point can be described using XML templates formatted as WSDL[5] definitions. So in figure 3, the Choice Points denoted could easily be implemented as web service driven components that provide control and selection within the implementation layer.

The Choice Points could also interact with a registry of definitions so that the complete behaviour can be externally configured and context driven. With such adaptability this delivers agile information flows based on business context.

2.1. Choice Point Implementation

The Choice Points have been described so far as abstract concepts. This section provides design details of the operation of Choice Points and their behaviors. To understand this we need to first collect the required Choice Point behaviors discussed so far above and summarize these:

- Allow inputs (facts) to determine outcomes (choices) based on rules

---

[5] Web Service Description Language, a W3C specification for describing web service points, their access and operations.

- Rules can be expressed and asserted non-procedurally with simple business-friendly constructs and syntax
- Choice Points can call Choice Points
- Assertion of facts and / or rules can be passed as inputs to a Choice Point
- Choice Points may inherited context details
- Decisions may be persisted for later process needs
- Choices can be a simple fixed set, or could be a dynamic set
- Choice Points are exposed as components of the architecture and not closed as inaccessible within a solution
- Choice Points can communicate via web services and messaging as needed
- Choice Points can hold the transient state of interactions

Next we consider the implementation constraints. The intention here is to provide a neutral definition of the BCM Choice Point mechanisms and their XML representations that implementers can then construct and integrate using popular rule engines. Since each application needs will vary it is important that implementers can choose to build a tailored sub-set while maintaining interoperability across Choice Points as the prime requirement.

Since Choice Points may interact themselves it is vital that the base functionality be established via the use of an open XML driven service with an API (application programming interface). Part of establishing this includes the ability to use a broad set of communications via WSDL definitions. Other OASIS technical specifications have already successfully implemented this approach, including the OASIS CAM specification. A further implementation need is that the Choice Point mechanism can be used by other OASIS specifications to provide dynamic context driven behaviors. Examples that have already been identified include: BPEL, BPSS, CAM, CPPA, UBL, and the CIQ specifications.

In order to construct a consistent XML driven API the following components are needed:

- Rule base and consistent decision mechanisms with supporting XML syntax
- Fact base and consistent representations in XML syntax for context
- State tracking and ability to assign globally unique thread IDs
- Query and Response action formats
- Change action formats
- Event handling formats
- Security support with audit trail within the Choice Point implementation

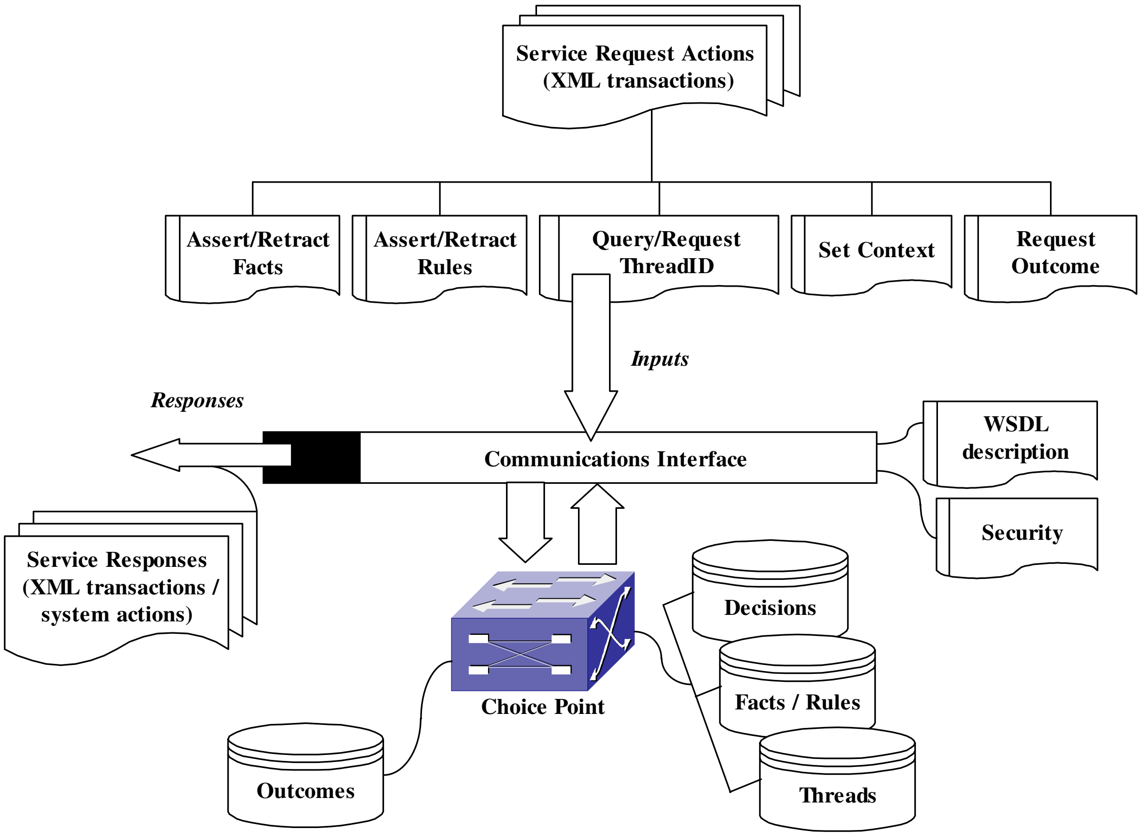
This summary is provided here, each of these items is expanded more thoroughly in the Choice Point technical specification itself[6]. The primary behaviors are listed first, while those behaviors likely to be optionally included in implementations are listed last.

Figure 5 depicts these components of the Choice Point implementation.

---

[6] See BCM technical specifications for these details.

Figure 5 – Choice Point rule engine implementation components

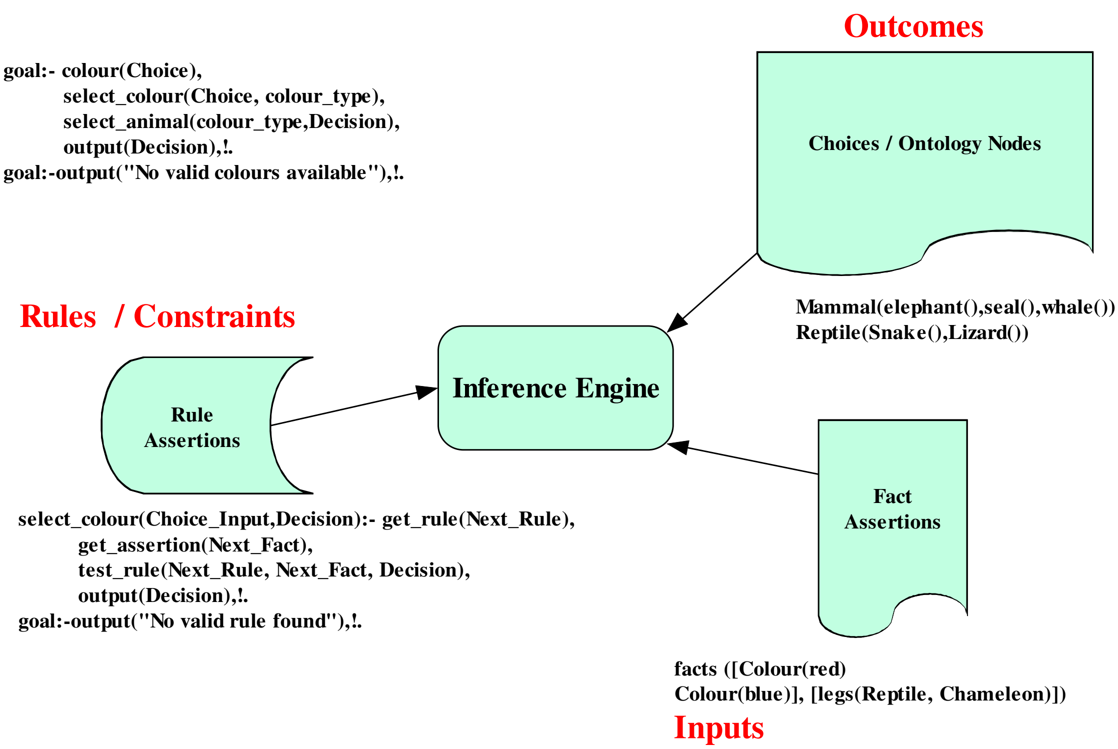


The Choice Point engine itself can have a variety of behaviors supported by the rule engine. Not all may be required, depending on the business application. This flexibility means that the Choice Point approach can be implemented directly using popular programming languages, without the need for a specialized rule engine, or alternately can be configured to use a rule agent. The varieties of anticipated common needs of these behaviors include:

- Fact assertion / retraction
- Rule assertion / retraction
- State tracking mechanism
- Simple case rule determination (select-when-otherwise)
- Solution determination via backtracking supported
- Solution determination via forward tracking supported
- Solution determination using constraint logic supported
- Storage of current state decision memory for later recall (decision threads)
- Decision testing support (if-then analysis)
- Audit trail and decision verification (why was this decision chosen?)
- Event handling support

To complete this section on Choice Point implementation figure 6 shows a possible configuration using a Prolog programming language based inference engine. Prolog has been used extensively for decision support implements and a wide variety of proven implementations are available. This example is not intended to be normative but merely to show the concepts behind implementing dynamic rule based decision processes. These mechanisms then require support via the XML formats and syntax of the Choice Point specification. It is therefore helpful to understanding those constructs and their behaviors.

Figure 6 – Example of decision rules processing



Referencing figure 6 above, the interface is shown in the “goal” section that controls the decision process. The WSDL interface to the Choice Point will need to expose support for such interactions. Similarly the “Rules / Constraints” will be implemented in XML syntax and a human friendly front-end provided that allows business users to create these. And then the facts and outcomes similarly will be input from a front-end and have XML formats for their creation and exchange. The implementer can then provide a bridge between their own internal Prolog syntax and the open Choice Point XML formats and syntax. As noted earlier, considerable work has already been done in this area of representation of rules logic using XML including such work as RuleML – Rules Markup Language and BRML - Business Rules Markup Language and others – and a complete list is available with links at: http://xml.coverpages.org/ruleML.html. Other noteworthy work is that done by the SHOE team – working on Simple HTML Ontology Extensions http://www.cs.umd.edu/projects/plus/SHOE/ . The need is to combine this earlier work with the Choice Point requirements to produce an implementation set that can deliver the needed behavior overall.

2.2. Summary and Next Steps

The BCM Choice Point approach provides a vital component for implementing agile information systems. With the advent of web service based Service Orientated Architectures this component is urgently required to ensure consistent implementations today. Furthermore the traditional e-Business systems interfaces within this model also need to transition their processes and content handling to support Choice Points as a means to deliver interoperability and adaptability.

While decision support systems in the past have implemented such techniques they have done so as closed systems. The opportunities that open rule-formats using XML together with interoperable communications brings is to remove the limitations of prior architectures and provide dynamic context driven implementation of enterprise systems.

This section of the BCM specifications is intended to facilitate this and form the basis for the scope of action of the Linking and Switching sub-committee (SC) of the BCM technical committee (TC).

It is anticipated that further liaison and outreach with other OASIS technical committees (TCs) will occur to refine requirements and the implementation model, and this process has already begun. Part of the deliverables for the sub-committee will include the creation of W3C WSDL models for Choice Points that will help other groups to understand the interface from their own specifications.

In parallel with these liaison efforts is the development of an initial Choice Points technical specification details (the Pareto Principle applies!) leading to prototyping using available rule engines and a demonstration using selected business scenarios.

Those interested in contributing to this work are encouraged to join the OASIS BCM TC and the Linking and Switching SC, more details on this are available from the OASIS website (http://www.oasis-open.org).

Notices