WS-CDL Primer

Date: 17th January 2005
Authors: Steve Ross-Talbot, Anthony Fletcher
Version: V0.1

Short Table of Contents

Table of Contents

1. Introduction

1.1 Prerequisites

This primer assumes that the reader has the following prerequisite knowledge:

- familiarity with XML (Extensible Markup Language (XML) 1.0 (Second Edition) [XML 1.0], XML Namespaces (Namespaces in XML [XML Namespaces]), WSDL (Web Services Description Language) 1.1 and 2.0;
- some familiarity with XML Schema (XML Schema Part 1: Structures [XML Schema: Structures] XML Schema Part 2: Datatypes [XML Schema: Datatypes]), SOAP (Simple Object Access Protocol) 1.2;
- familiarity with basic Web services concepts such as Web service, client, and the purpose and function of a Web service description. (For an explanation of basic Web services concepts, see Web Services Architecture [WS Architecture] Section 1.4 and Web Services Glossary [WS Glossary] glossary. However, note the Web Services Architecture document uses the slightly more precise terms "requester agent" and "provider agent" instead of the terms "client" and "Web service" used in this primer.)

No previous experience with WS-CDL is assumed.

1.2 Structure of this Primer

Section 2 presents an overview of WS-CDL. It segments the language and describes what it can be used for and presents the business benefits that can be gained from using WS-CDL. Finally a methodology is described which provides a guide as to how to build a choreography.

Section 3 presents a hypothetical use case involving a market in which buyers and sellers and supporting roles enact their business. It proceeds step-by-step based on the methodology described in Section 2 through the development of this simple example.

Section 4 introduces more advanced topics that deal with notions of business collaborations and failure as well as complex ordering constraints that can be described in WS-CDL.

Section 5 describes the relationships that WS-CDL has to some of the other standards and technologies. This includes code generation to Java, .NET and BPEL4WS, the relationship through the use of extensions to Policy based technology and Service Discovery.

1.3 Notational Conventions

The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC-2119 [RFC2119].

The following namespace prefixes are used throughout this document:

<Table>

This specification uses an informal syntax to describe the XML grammar of a WS-CDL document:

- The syntax appears as an XML instance, but the values indicate the data types instead of values.
- Characters are appended to elements and attributes as follows: "?" (0 or 1), "*" (0 or more), "+" (1 or more).
- Elements names ending in ". . ." (such as <element. . ./> or <element. . .>) indicate that elements/attributes irrelevant to the context are being omitted.
- Grammar in bold has not been introduced earlier in the document, or is of particular interest in an example.
- <-- extensibility element --> is a placeholder for elements from some "other" namespace (like ##other in XSD).
- The XML namespace prefixes (defined above) are used to indicate the namespace of the element being defined.
- Examples starting with <?xml contain enough information to conform to this specification; other examples are fragments and require additional information to be specified in order to conform.

An XSD is provided as a formal definition of WS-CDL grammar (see Section 11 of the Web Services Choreography Description Language Version 1.0 W3C Working Draft 17 December 2004).

2. An Overview of WS-CDL

The Web Services Choreography Description Language (WS-CDL) is an XML-based language that can be used to describe the common and collaborative behavior of multiple parties who need to interact in order to achieve some goal. WS-CDL describes this behavior from a global or neutral perspective rather than from the perspective of any one party.

2.1 Using WS-CDL

WS-CDL is not an executable language, hence the term “Description” in it’s name. It is a language that can be used to unambiguously describe business collaborations and their protocols within and across domains of control.

In the case of the former it can be used to describe the internal workflows within a domain that involves multiple end-points/services that constitute collaborative behavior. The value in so doing is to ensure continued conformance of services to an agreed choreography description and to guarantee interoperability of services through an agreed choreography description. This is no more than describing a business protocol that defines a collaboration between services. You can think of it as a way of ensuring services are well behaved with respect to the goals that you want to achieve within your domain.

In the case of the latter it can be used to describe the business protocols across domains such as the ordering of message exchanges that govern vertical protocols such as fpML, FIX, TWIST and SWIFT. These protocols have some form of XML data format definition and then go on to describe the ordering of message exchanges using a combination of prose and UML sequence diagrams. What WS-CDL provides in an unambiguous way of describing the ordering of message exchanges and in so doing ensure that the end points that participate in business collaborations based on such vertical standards can be guaranteed to conform to the choreography description. You can think of it as a way to ensure that participants are well behaved with respect to their common goals across domains.

2.2 The Business Case (reword)

WS-CDL can be used to ensure interoperability within and across domains of control. In so doing WS-CDL can remove problems of interoperability and so deliver business solutions within and across domains of control

that are more reliable with less downtime due to the errors in encoding a business protocol.

WS-CDL can be used to ensure that the total cost of software systems in a distributed environment, within a domain of control and across the world-wide-web is lowered by guaranteeing that the services that participate in a choreography are well behaved on a continuing basis.

Both of these benefits translate into more up-time and so increase top line profits and at the same time they translate into less testing time and so reduce cost of delivery which decreases bottom line costs.

2.3 The Structure of WS-CDL

WS-CDL is a layered language. It provides different levels of expressibility to describe a choreography. These levels are shown below (a more complete picture is provided by the UML description of WS-CDL in the appendix):

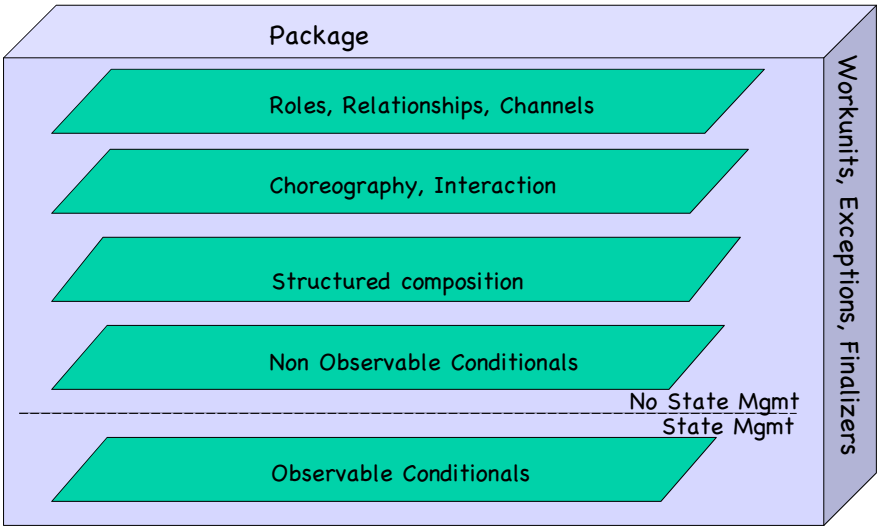

Figure 1: The layering of WS-CDL

At the top most level for any WS-CDL there is a package which contains all other things. All choreographies described in WS-CDL will include as a minimum a set of Roles that are defined as some sort of behavior (i.e. a WSDL description), Relationships between those roles, Channels used by roles to interact and a Choreography block that uses the channels to describe Interaction. What the choreography describes at this level is a basic set of typed and unambiguous connections that enable the various roles to collaborate in order to achieve some common goal.

Choreographies that are described using only these features will be uninteresting as there will be few ordering rules.

Adding further ordering rules through Structured composition allows Interactions and Choreographies (which are just logical groupings of interactions) to be combined into sequences, parallel activities and so on.

Adding Non-Observable Conditionals makes it is possible to model branching based on observing changes in the interactions that occur – for example one might observe an exchange between a buyer and a seller which is said to be terminated when a "completed" interaction is observed.

At this point it is not necessary to perform any explicit state management at the roles that are interacting because we have not needed to express any observable conditions. By this we mean that none of roles used in choreographies at this level have any notion of shared state, rather they observe interactions that are visible and use the observations to determine their state with respect to the other roles.

Some business protocols are defined with specific business rules visible. These constitute shared knowledge between the roles concerned, for example we might terminate an order completion between a buyer and a seller when we calculate that the items delivered match the original order. The business rule in this example is the shared constraint that buyer_quantity equals completion_quantity. At some level the roles must have some shared knowledge of both variables and their values. When business rules of this nature become part of the business protocol such Observable Conditionals can be added into a choreography and this now implies state management is needed.

State management requires an amount of machinery to ensure that state is known globally when needed or at least between roles when needed. The machinery required to deliver this in practice is some coordination mechanism that will ensure data is delivered to all roles that require it.

2.4 Methodology

Given that WS-CDL is a language that described interaction between roles it is natural to look first for the roles that will interact. This is the starting point for any choreography. The methodology described herein documents a process that can be used to guide the user in creating a choreography.

1. Identify the role that will interact
2. Defining the relationships
   - between the roles that will communicate
3. Define the channels
   - that will be used to connect the roles together
   - directionality: Who receives and who initiates
4. Group the roles into participants
5. Sketch the choreographies (a grouping of high level logical interactions)
6. Define the interactions
   - Channel variables, Information Types
7. Define state management
   - Records, variables
8. Refine the choreographies (by grouping into finer grain interactions)
   - Work Units, Finalization

This process is not the only process that can be used to define choreographies but we have found it generally consistent in using the WS-CDL to describe complex business protocols.