Re-submission of the Instance-Set Use Case

I have noticed in WS-CHOR 30/9/2003 minutes that there is some interest in the Instance-Set Use Case. Hoping to provide the required clarification, I am now resubmitting my proposal.

Regards

Marco Adragna
marco.adragna@kellogg.ox.ac.uk

Index

1. Actors

1.1. Parent

A Parent is a Web Service allowing users to perform operations that relate to a homogenous set of Instances. Such Instances represent “Examples Of” a known Classifier associated with the Parent.

Typical operations that a Parent MAY support are:
a) Addition of a new Instance to the Set.
b) Lookup of an existing Instance using names, business data, primary keys or other information.
Both the Addition and the Lookup operations return Users an “Identifier” of the Instance.
Such “Identifier” is typically a WSDL, a URI , a correlation-id or a custom web service reference.

In relation to the addition of a new instance, the Parent can adopt three externally observable behaviours:

Factory) Behave as a Factory Method, actually creating new Instances.

Naming) Support the logical addition to the Instance-Set of an existing web service. The Classifier associated with the Parent MUST match the WSDL Interface of the existing web service that is being added to the Instance-Set. Such newly added Instance is binded with a unique name, which can then be used to found the Instance via look-up operation.

Singleton) Behave as a Singleton with an X number of instances, providing a unique point of access to those X instances via Lookup methods and forbidding the creation of new instances.

1.2. Instance

All Instances belonging to a given Instance-Set are Web Services described by the same WSDL Interface. Such common Interface is the **Classifier** of the given set of Instances.
Every Instance of a set has an individual state and a unique name that differentiates it from the others.

1.3. User

Is a software system that wishes to consume web services that are “Examples Of “a known Classifier.

2. Description

2.1. Motivation for Usage

The Instance-Set use case is needed in web service software systems that have classifier/instance relationships in their behavioural requirement. Parent and Instances need to communicate to Users which message sequences are legal. Users need to know how to correctly create, find and interact with Web Services that represent Instances of a known Classifier.

2.2. Example of usage: e-learning and the Lesson Web Service

The e-learning company LivePro wants to develop a distributed software system to enables Consultant to sell Lessons to Students in needs of their expertise.
A Consultant can create a new Lesson and define a time for its commencement.
Alternatively, a Student can request a custom-made Lesson and wait for a Consultant that can teach it.
Every Lesson is recorded and can be bought again by another Student, unless the Consultant who taught it decides to delete it.

In order to offer a fully featured product, LivePro Company needs to integrate its software with third-party providers, which offer cutting-edge Virtual Classroom environments and Payment Processing facilities. Those providers have made available such functionalities via web services. Those and other considerations have led the LivePro board to decide in favour of Web Service technology.

LivePro technical staff is left with the challenge of how to cleanly design a Lesson Web Service allowing clients to create and interact with Lesson Instances such as ItalianLanguageLesson, AdvancedMathLesson and so forth. Fortunately a senior developer has heard about the Instance-set Choreography and informs the rest of the staff about it. Using the WS choreography language is possible to express in a standard and machine-readable fashion how Clients should create and use Lesson Instances. Two separate WSDL Interfaces are needed for such purpose:

- LessonParent: allows creating, finding and interacting with Lesson Instances.
- LessonClassifier: define all the operation/messages that are typical of a Lesson and are used to perform booking, rescheduling and so forth.

3. Preconditions

The User MUST be aware of the Instance-Set choreography

4. Flow Of Events

4.1. Basic Flow

A user might exchange a sequence of messages that logically
begins with obtaining an identifier of a new Instance from its Parent and
ends either when the individual state of that Instance is reclaimed
or when the User's copy of the Instance Identifier is destroyed.

4.2. Alternate Flow

A user might exchange a sequence of messages that logically
begins with obtaining an identifier of an existing Instance from its Parent and
ends either when the individual state of that Instance is reclaimed
or when the User's copy of the Instance Identifier is destroyed.
Such existing identifier might be obtained via LookUp operations provided by the Parent.

5. Notes / Issues

It needs to be stressed that the classifier-instance relationship is not in any way an implementation-related property only needed in object-oriented systems. In fact, the aggregation/part and the classifier/instance relationships are two fundamental logical constructs that Requirement Engineering uses to describe any software system.

With "User's copy of the Instance Identifier" we typically refer to a local copy of the WSDL document of a remote Instance, to any additional value or custom reference that identify that particular instance and to any stub/proxy object built using them.

In this document we use the words “Instance Of” and “Example Of” interchangeably with the same meaning expressed by sentences such as “Apple, strawberry and lemon are Example Of fruit” and “Dog and wolf are Example Of animals”.

In this document we often refer to “the operation X is performed” and “support the operation X” that in this context have the meaning of “the SOAP rpc-style message X is exchanged” and “understand the SOAP rpc-style message X”.

When the “Individual State of an Instance is reclaimed” the history of interaction of such Instance is lost.
The Instance returns to a state that corresponds to a newly created Instance.
The reclaim operation can be initiated either by the User or by the Parent or by the Instance itself depending on the specific choreography.

A user might possess an Identifier to an Instance of which the individual state has been reclaimed.
A fault condition MUST be generated by messages being sent to such Instance.
Users MUST be able to handle such fault.