The Workflow Management Coalition Specification

Workflow Management Coalition Workflow Standard

Workflow Process Definition Interface -- XML Process Definition Language

Document Number WFMC-TC-1025
Document Status – 1.0 Final Draft

October 25, 2002
Version 1.0

Workflow Management Coalition
2436 N. Federal Highway #374
Lighthouse Point, Fl 33064
USA

Tel: +1 954 782 3376
Fax: +1 954 782 6365
Email: wfmc@wfmc.org
WWW: http://www.wfmc.org

First printing July 2002

Table of Content

1. CHANGE HISTORY 3

2. AUDIENCE 4

3. PURPOSE 4

4. INTRODUCTION 4

4.1. CONFORMANCE 5
4.2. REFERENCES 5

5. OVERVIEW OF PROCESS DEFINITION INTERCHANGE 6

5.1. APPROACHES TO PROCESS DEFINITION INTERCHANGE 6

6. META-MODEL 8

6.1. ENTITIES OVERVIEW 8
6.1.1. Workflow Process Definition 8
6.1.2. Workflow Process Activity 8
6.1.3. Transition Information 9
6.1.4. Workflow Participant Declaration 9
6.1.5. Resource Repository 9
6.1.6. Workflow Application Declaration 9
6.1.7. Workflow Relevant Data 10
6.1.8. System and Environmental Data 10
6.1.9. Data Types and Expressions 10
6.2. PROCESSES AND PACKAGES 10
6.3. PROCESS META-MODEL 12
6.4. PACKAGE META-MODEL 12
6.4.1. Process Repository 13
6.4.1.1. Redefinition and Scope 13
6.5. ELEMENTS OVERVIEW 14
6.5.1. Vendor or User specific Extensions 15
6.5.1.1. Extended Attributes 15
6.5.1.2. Extended parameter mapping 15

7. XML PROCESS DEFINITION LANGUAGE 16

7.1. ELEMENTS COMMON FOR MULTIPLE ENTITIES 16
7.1.1. Extended Attributes 16
7.1.2. Formal Parameters 16
7.1.2.1. Parameter passing semantics 17
7.1.2.2. Concurrency semantics 17
7.1.2.3. Formal-actual parameter mapping 17
7.1.3. External Reference 18
7.1.3.1. Web Services 18
7.2. PACKAGE DEFINITION 19
7.2.1. Package definition Header 19
7.2.2. Redefinable Header 20
7.2.3. Conformance Class Declaration 22
7.2.4. Script 22
7.2.5. External Package Reference 23
7.3. WORKFLOW APPLICATION DECLARATION 23
7.3.1. Invocation Parameters 24
7.4. WORKFLOW PROCESS DEFINITION 24
7.4.1. Workflow Process Definition Header 25
7.4.2. Workflow Process Redefinable Header 27
7.4.3. Activity Set 28

7.5. WORKFLOW PROCESS ACTIVITY 29
7.5.1. Route Activity *32*
7.5.2. Block Activity *32*
7.5.3. Execution Control Attributes *32*
7.5.4. Implementation Alternatives *33*
7.5.4.1. No Implementation 33
7.5.4.2. Tool 34
7.5.4.3. Subflow 34
7.5.5. Performer Relationship *35*
7.5.6. Deadline *36*
7.5.7. Simulation Information *37*
7.5.8. Transition Restrictions *38*
7.5.8.1. Join 38
7.5.8.2. Split 39
7.5.9. Conformance Classes *40*
7.6. TRANSITION INFORMATION 40
7.6.1. Condition *42*
7.6.1.1. Exception Conditions 42
7.7. WORKFLOW PARTICIPANTS 43
7.7.1. Participant Entity Types *44*
7.8. WORKFLOW RELEVANT DATA 44
7.9. DATA TYPES 46
7.9.1. Basic Data Types *46*
7.9.2. Complex Data Types *47*
7.9.2.1. Schema Type 47
7.9.2.2. Record Type 48
7.9.2.3. Union Type 48
7.9.2.4. Enumeration Type 48
7.9.2.5. Array Type 49
7.9.2.6. List Type 49
7.9.3. Declared Data Types *49*
7.9.3.1. Type Declaration 49
7.9.3.2. Declared Type 50

8. SAMPLE WORKFLOW 52

8.1. THE PROCESSES 52
8.1.1. The EOrder Main Process *52*
8.1.2. The CreditCheck Subprocess *53*
8.1.3. The FillOrder Subprocess *53*
8.2. TYPE DECLARATIONS 54
8.3. EXTENDED ATTRIBUTES 56
8.4. EXTERNAL REFERENCES 56
8.5. SAMPLE XPDL 57

9. XPDL SCHEMA 73

10. FIGURES AND TABLES 84

10.1. FIGURES 84
10.2. TABLES 84

1. Change History

Version 1.0 – Editor: Roberta Norin (robertan@attbi.com).

- Added processContent="lax" attribute to specification of SchemaType, ExtendedAttribute, and Xpression.
- Changed URI references in the sample workflow to point to documents on the wfmc.org web site.
- Changed SubFlow Id's in sample workflow to refer to Workflow Process by Id rather than Name.

Version 0.10 – Editor: Roberta Norin (robertan@attbi.com). Contributors: Seth Osher (Intuitive Products International Corp.) and Robert Shapiro (Cape Visions).

- Removed InlineBlock and BlockName elements from schema, from content of TransitionRestriction. Removed BlockName element, and from specification.
- Added ActivitySets and BlockActivity to schema and to specification
- Removed maxOccurs attribute from Activities in schema.
- Added Deadline element to schema and specification.
- Integrated Deadline into Sample Workflow.
- Replace meta model references with UML diagrams.
- Add section describing how to specify web services in xpdl.

Version 0.09 – Editor: Roberta Norin (robertan@attbi.com)

- Added Chapter 8 – Sample Workflow.

Version 0.08 – Editor: Roberta Norin (robertan@attbi.com) Contributor: Mike Gilger (Identitech)

- Removed DataTypes from WorkflowProcess.
- Added BOOLEAN and PERFORMER to BasicType.
- Removed PlainType Element from Schema.
- Removed reference to PlainType from DataTypes
- Added Script Element.
- Add reference to Script Element to Package.
- Removed left over references to LOOP in Conformance class and transition discussions.
- Completed Condition table in Section 7.6.1.
- Added discussion of loops to Section 7.6.

Version 0.07 – Editor: Roberta Norin (robertan@attbi.com)

- Reconfigured the DataTypes element (which was not being used) to be an xsd:group that contained references to all the data types; refer to this group wherever the list of data types was repeated.
- Edited the Data Types Section to emphasize the use of SchemaType to define complex data, to clarify the use of TypeDeclarations, and to take advantage of the DataTypes group simplification.
- Moved SchemaType discussion to be under Complex Data types.

Version 0.06 – Editor: Roberta Norin (robertan@attbi.com)

- Added AccessLevel attribute to WorkflowProcess
- Added ExternalReference to Participant
- Removed Loop Implementation from WorkflowActivity/Implementation
- Removed Loop Element.

- Removed Loop Attribute from Transition.
- Removed Loop Activity from Figure 7.1.
- Added TargetNamespace designation to schema (.xsd). Used xpdl namespace prefix in references to xpdl elements.
- Added SchemaType and ExternalReference to all lists of Data Types.
- Rearranged all lists of Data Types so that the old way of declaring complex types (array, record, etc.) are last in list of choices.

Version 0.05 – Editor: Roberta Norin (robertan@attbi.com)

- Removed (redundant) discussion of Parameters under WorkflowProcess Activity and folded it into Formal Parameters section 7.1.2.
- Completed missing text in tables in Chapter 7.

Version 0.04 – Editors: Mike Marin (mmarin@filenet.com) and Roberta Norin (rnorin@apengines.com)

- Incorporates modifications discussed in the May WfMC meeting.
- This version uses a XML Schema instead of a DTD to describe XPDL
- Added External References, which provides a way to interact with web services (WSDL) and other external definitions.
- Added Schema Type, that allows for the definition of complex types by using a XML schema.
- Introduced the concept of exception for routing.

Versions 0.02/0.03 – Editor: Mike Marin (mmarin@filenet.com)

- Changes based on review by working group 1 during the New York meeting May 3 and 4 of 2001. This version has significant input from Roberta Norin (AP Engines), Robert Shapiro (Cape Visions), and all the other participants of the working group during the New York meeting.

Version 0.01 – Editor: Mike Marin (mmarin@filenet.com)

- Initial Version.

2. Audience

The intended audience for this document is primarily vendor organizations who seek to implement the XML Process Definition Language (XPDL) of the Workflow Management Coalition (WfMC). It may also be of interest to those seeking to assess conformance claims made by vendors for their products. Comments should be addressed to the Workflow Management Coalition.

3. Purpose

The WfMC has identified five functional interfaces to a workflow service as part of its standardization program. This specification forms part of the documentation relating to "Interface one" - supporting Process Definition Import and Export. This interface includes a common meta-model for describing the process definition (this specification) and also an XML schema for the interchange of process definitions.

4. Introduction

A variety of different tools may be used to analyse, model, describe and document a business process. The workflow process definition interface defines a common interchange format, which supports the transfer of workflow process definitions between separate products.

The interface also defines a formal separation between the development and run-time environments, enabling a process

definition, generated by one modelling tool, to be used as input to a number of different workflow run-time products.

A workflow process definition, generated by a build-time tool, is capable of interpretation in different workflow run-time products. Process definitions transferred between these products or stored in a separate repository are accessible via that common interchange format.

To provide a common method to access and describe workflow definitions, a workflow process definition meta-data model has been established. This meta-data model identifies commonly used entities within a process definition. A variety of attributes describe the characteristics of this limited set of entities. Based on this model, vendor specific tools can transfer models via a common exchange format.

One of the key elements of the XPDL is its extensibility to handle information used by a variety of different tools. XPDL may never be capable of supporting all additional information requirements in all tools. Based upon a limited number of entities that describe a workflow process definition (the "Minimum Meta Model"), the XPDL supports a number of differing approaches.

One of the most important elements of XPDL is a generic construct that supports vendor specific attributes for use within the common representation. We recommend that any missing attributes be proposed to the WfMC interface one workgroup for inclusion in future releases.

This document describes the meta-model, which is used to define the objects and attributes contained within a process definition. The XPDL grammar is directly related to these objects and attributes. This approach needs two operations to be provided by a vendor:

- Import a workflow definition from XPDL.
- Export a workflow definition from the vendor's internal representation to XPDL.

A vendor can use a XSL style sheet to comply with those two operations.

All keywords and terms used within this specification are based upon the WfMC Glossary.

For the purpose of this document, the terms process definition, business process model, and workflow model are all considered to represent the same concept, and therefore, they are used interchangeably.

4.1. Conformance

A vendor can not claim conformance to this or any other WfMC specification unless specifically authorised to make that claim by the WfMC. WfMC grants this permission only upon the verification of the particular vendor's implementation of the published specification, according to applicable test procedures defined by WfMC.

Conformance for process definition import / export is essentially based upon conformance to the XPDL grammar. However, there is a mandatory minimum set of objects, as specified within this document, which must be supported within XPDL. But, given the wide variation of capabilities in modelling tools, it is reasonable to assume that an individual tool might conform to this specification but not be able to swap complete definitions with all other conforming products. A product that claims conformance must generate valid, syntactically correct XPDL, and must be able to read all valid XPDL.

4.2. References

The following documents are associated with this document and should be used as a reference.

General background information:

WfMC Terminology & Glossary (WfMC-TC-1011)

WfMC Reference Model (WfMC-TC-1003)

WfMC API specifications, which include process definition manipulation APIs:

WfMC Client Application API Specifications (WAPI) (WfMC-TC-1009)

WfMC Process Definition Interchange – Process Model (WfMC-TC-1016-P)

Workflow process interoperability, used to support process invocation on a remote workflow service:

Workflow Interoperability - Abstract Specifications (WfMC-TC-1012)

Interoperability - Internet E-mail MIME Binding (WfMC-TC-1018)

Accompanying documents:

The Resource Model (Organizational Model: WfMC TC-1016-O)

5. Overview of Process Definition Interchange

A Process Definition is defined as:

> *The representation of a business process in a form that supports automated manipulation, such as modeling, or enactment by a workflow management system. The process definition consists of a network of activities and their relationships, criteria to indicate the start and termination of the process, and information about the individual activities, such as participants, associated IT applications and data, etc. (WfMC Glossary - WfMC-TC-1011)*

The process definition provides an environment for a rich description of a process that can be used for the following,

- Act as a template for the creation and control of instances of that process during process enactment.
- For simulation and forecasting.
- As a basis to monitor and analyse enacted processes.
- For documentation, visualization, and knowledge management.

The process definition may contain references to subflow, separately defined, which make up part of the overall process definition.

An initial process definition will contain at least the minimal set of objects and attributes necessary to initiate and support process execution. Some of these objects and attributes will be inherited by each created instance of the process.

The WfMC Glossary also contains descriptions of, and common terminology for, the basic concepts embodied within a process definition such as activities, transitions, workflow relevant data and participants, etc.

5.1. Approaches to Process Definition Interchange

This specification uses XML as the mechanism for process definition interchange. XPDL forms a common interchange standard that enables products to continue to support arbitrary internal representations of process definitions with an import/export function to map to/from the standard at the product boundary.

A variety of different mechanisms may be used to transfer process definition data between systems according to the characteristics of the various business scenarios. In all cases the process definition must be expressed in a consistent form, which is derived from the common set of objects, relationships and attributes expressing its underlying concepts.

The principles of process definition interchange are illustrated in Figure 5-1: The Concept of the Process Definition Interchange.

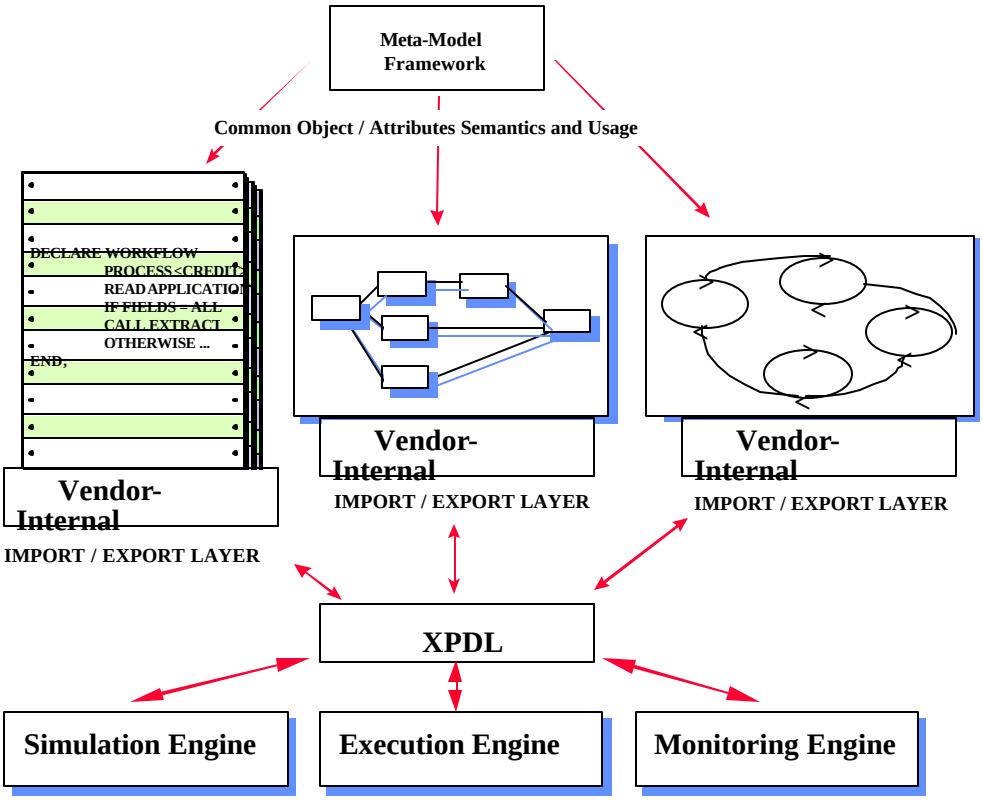

Figure 5-1: The Concept of the Process Definition Interchange

6. Meta-Model

The Meta-Model describes the top-level entities contained within a Process Definition, their relationships and attributes (including some which may be defined for simulation or monitoring purposes rather than for enactment). It also defines various conventions for grouping process definitions into related process models and the use of common definition data across a number of different process definitions or models.

The top-level entities are shown in the following figure:

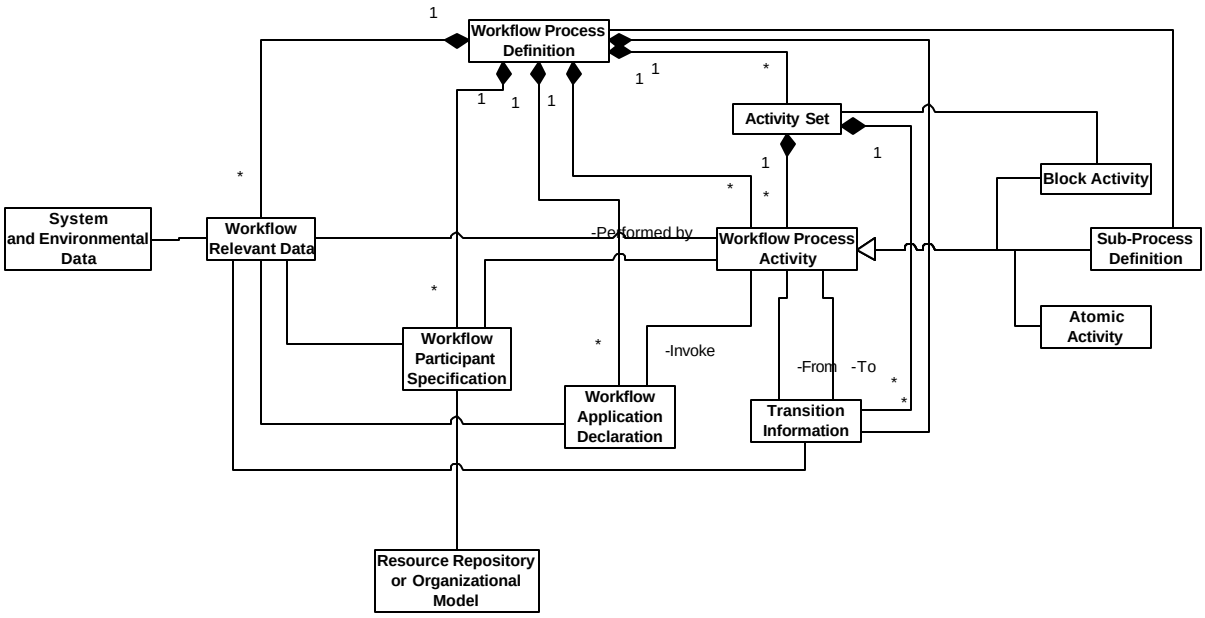

Figure 6-1: Meta-Model top-level entities

For each of the above entities, there is an associated set of properties, which describe the characteristics of the entity. The following sections describe these entities and properties in more detail.

6.1. Entities Overview

The meta-model identifies the basic set of entities used in the exchange of process definitions. The top-level entities are as follows:

6.1.1. Workflow Process Definition

The Process Definition entity provides contextual information that applies to other entities within the process. It is a container for the process itself and provides information associated with administration (creation date, author, etc.) or to be used during process execution (initiation parameters to be used, execution priority, time limits to be checked, person to be notified, simulation information, etc.).

6.1.2. Workflow Process Activity

A process definition consists of one or more activities, each comprising a logical, self-contained unit of work within the process. An activity represents work, which will be processed by a combination of resource (specified by participant assignment) and/or computer applications (specified by application assignment). Other optional information may be

associated with the activity such as an information on whether it is to be started / finished automatically by the workflow management system or its priority relative to other activities where contention for resource or system services occurs. Usage of specific workflow relevant data items by the activity may also be specified. The scope of an activity is local to a specific process definition (although see the description of a subflow activity below).

An activity may be a subflow - in this case it is a container for the execution of a (separately specified) process definition, which may be executed locally within the same workflow service, or (possibly using the process interoperability interface) on a remote service. The process definition identified within the subflow contains its own definition of activities, internal transitions, resource, and application assignments (although these may be inherited from a common source). In- and out-parameters permit the exchange of any necessary workflow relevant data between calling and called process (and, where necessary, on return).

An activity may be a block activity that executes an activity set, or map of activities and transitions. Activities and transitions within an activity set share the name space of the containing process.

Finally, a dummy activity is a skeletal activity, which performs no work processing (and therefore has no associated resource or applications), but simply supports routing decisions among the incoming transitions and/or among the outgoing transitions.

6.1.3. Transition Information

Activities are related to one another via flow control conditions (transition information). Each individual transition has three elementary properties, the from-activity, the to-activity and the condition under which the transition is made. Transition from one activity to another may be conditional (involving expressions which are evaluated to permit or inhibit the transition) or unconditional. The transitions within a process may result in the sequential or parallel operation of individual activities within the process. The information related to associated split or join conditions is defined within the appropriate activity, split as a form of “post activity” processing in the from-activity, join as a form of “pre-activity” processing in the to- activity. This approach allows the workflow control processing associated with process instance thread splitting and synchronization to be managed as part of the associated activity, and retains transitions as simple route assignment functions. The scope of a particular transition is local to the process definition, which contains it and the associated activities.

More complex transitions, which cannot be expressed using the simple elementary transition and the split and join functions associated with the from- and to- activities, are formed using dummy activities, which can be specified as intermediate steps between real activities allowing additional combinations of split and/or join operations. Using the basic transition entity plus dummy activities, routing structures of arbitrary complexity can be specified. Since several different approaches to transition control exist within the industry, several conformance classes are specified within XPDL. These are described later in the document.

6.1.4. Workflow Participant Declaration

This provides descriptions of resources that can act as the performer of the various activities in the process definition. The particular resources, which can be assigned to perform a specific activity, are specified as an attribute of the activity, participant assignment, which links the activity to the set of resources (within the workflow participant declaration) which may be allocated to it. The workflow participant declaration does not necessarily refer to a human or a single person, but may also identify a set of people of appropriate skill or responsibility, or machine automata resource rather than human. The meta-model includes some simple types of resource that may be defined within the workflow participant declaration.

6.1.5. Resource Repository

The resource repository accounts for the fact that participants can be humans, programs, or machines. In more sophisticated scenarios the participant declaration may refer to a resource repository, which may be an Organizational Model in the case of human participants. Note that this specification does not define or require a resource repository.

6.1.6. Workflow Application Declaration

This provides descriptions of the IT applications or interfaces which may be invoked by the workflow service to support, or wholly automate, the processing associated with each activity, and identified within the activity by an

application assignment attribute (or attributes). Such applications may be generic industry tools, specific departmental or enterprise services, or localized procedures implemented within the framework of the workflow management system. The workflow application definition reflects the interface between the workflow engine and the application or interface, including any parameters to be passed.

6.1.7. Workflow Relevant Data

This defines the data that is created and used within each process instance during process execution. The data is made available to activities or applications executed during the workflow and may be used to pass persistent information or intermediate results between activities and/or for evaluation in conditional expressions such as in transitions or participant assignment. Workflow relevant data is of particular type. XPDL includes definition of various basic and complex data types, (including date, string, etc.) Activities, invoked applications and/or transition conditions may refer to workflow process relevant data.

6.1.8. System and Environmental Data

This is data which is maintained by the workflow management system or the local system environment, but which may be accessed by workflow activities or used by the workflow management system in the evaluation of conditional expressions in the same way as workflow relevant data.

6.1.9. Data Types and Expressions

The meta-model (and associated XPDL) assumes a number of standard data types (string, reference, integer, float, date/time, etc.); such data types are relevant to workflow relevant data, system or environmental data or participant data. Expressions may be formed using such data types to support conditional evaluations. Data types may be extended using an XML schema or a reference to data defined in an external source.

6.2. Processes and Packages

As indicated in the diagram above, the process model includes various entities whose scope may be wider than a single process definition. In particular the definitions of participants, applications and workflow relevant data may be referenced from a number of process definitions. The meta-model assumes the use of a common process definition repository, associated with the workflow management system, to hold the various entity types comprising the process definition. Within the repository itself and to support the efficient transfer of process definition data to/from the repository, the concept of a package is introduced, which acts as a container for the grouping of common data entities from a number of different process definitions, to avoid redefinition within each individual process definition.

The package provides a container to hold a number of common attributes from the workflow process definition entity (author, version, status, etc.). Each process definition contained within the package will automatically inherit any common attributes from the package, unless they are separately re-specified locally within the process definition

Within a package, the scope of the definitions of some entities is global and these entities can be referenced from all workflow process definitions (and associated activities and transitions) contained within the package. Those entities are:

- Workflow participant specification
- Workflow application declaration, and
- Workflow relevant data

The package reference allows the use within the package or its contained objects of references to top-level entities in the referenced external package:

- Process ids for subflow reference
- Workflow participant specifications
- Workflow application declarations

Conventions on name and identifier management across different packages within the same repository address space to achieve any necessary global uniqueness are for user/vendor definition. The assumed convention during process

enactment is that name reference searches follow the sequence:

- Process ids - firstly within the same model (including any references to process definitions for remote execution on a different service), then within any externally referenced model
- Applications / participants - firstly within the same model, then within any externally referenced model

Workflow relevant data naming must be unique within a package; where such data is passed between processes as parameters the convention at this version of specification is that copy semantics will be used. R esponsibility rests with process designers / administrators to ensure consistent name / data type usage within process definitions / models to support subflow operations (including any required remote process interoperability).

6.3. Process Meta-Model

The met a-model identifies the basic set of entities and attributes for the exchange of process definitions. For a Process Definition the following entities must be defined, either explicitly at the level of the process definition, or by inheritance directly or via cross reference from a surrounding package:

- Workflow Process Activity
- Transition Information
- Workflow Participant Specification
- Workflow Application Declaration
- Workflow Relevant Data

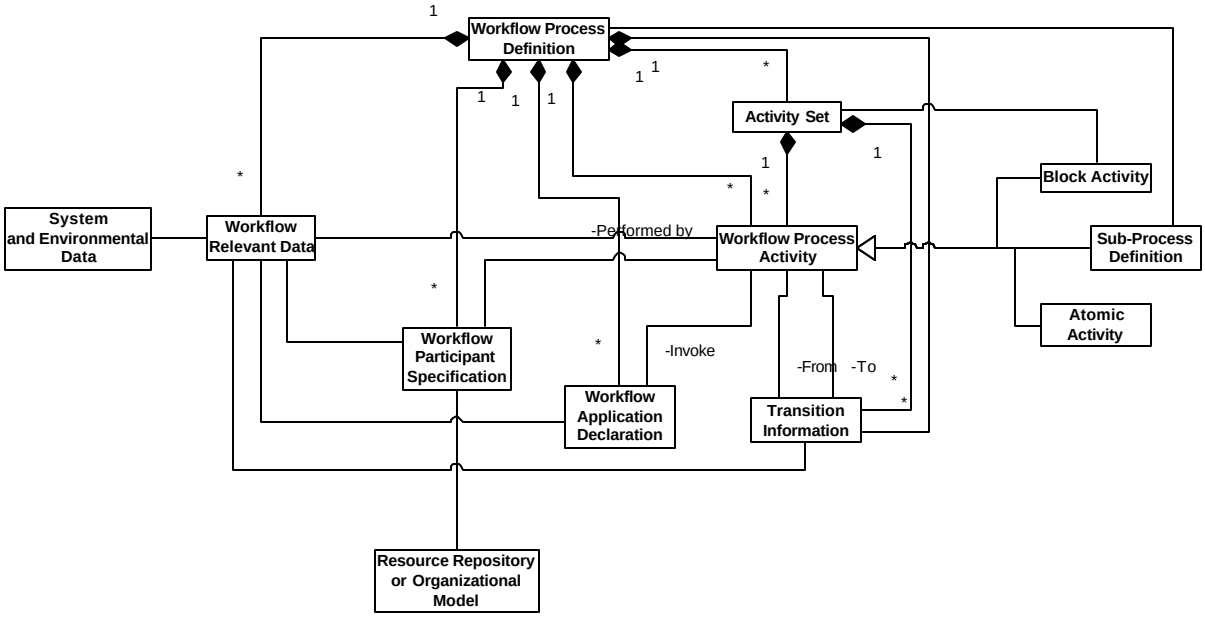

Figure 6-3: Workflow Process Definition Meta Model

These entities contain attributes that support a common description mechanism for processes. They are described in the subsequent document sections.

6.4. Package Meta-Model

Multiple process definitions are bound together in a model definition. The Package acts as a container for grouping together a number of individual process definitions and associated entity data, which is applicable to all the contained process definitions (and hence requires definition only once). The Package meta-model contains the following entity types:

- Workflow Process Definition
- Workflow Participant Specification
- Workflow Application Declaration
- Workflow Relevant Data

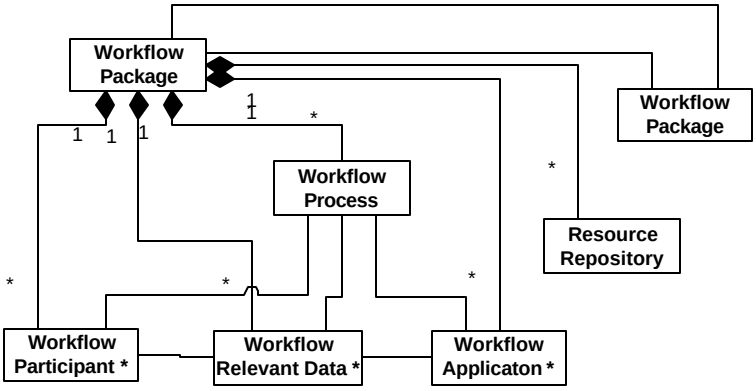

Figure 6-5: Package Definition Meta Model

The meta-model for the Package identifies the entities and attributes for the exchange, or storage, of process models. It defines various rules of inheritance to associate an individual process definition with entity definitions for participant specification, application declaration and workflow relevant data, which may be defined at the package level rather than at the level of individual process definitions.

The Package Definition allows the specification of a number of common process definition attributes, which will then apply to all individual process definitions contained within the package. Such attributes may then be omitted from the individual process definitions. (If they are re-specified at the level of an individual process definition this local attribute value takes precedence over the global value defined at the package level.

6.4.1. Process Repository

The process definition import/export interface is assumed to operate to/from a workflow definition repository of some form associated with the workflow management system. The import/export interface is realized by the transfer of files containing XPDL into or out of such repository. This interface specification allows the import or export of process definition data at the level of individual process definitions and packages.

The internal interface between the repository and workflow control functions is specific to individual vendor products and does not form part of this standard. It is assumed that separation is provided (for example by version control) between repository usage as a static repository (for persistent, ongoing storage of process definition data) and any dynamic usage (for managing changes to the process execution of extant process instances).

The local storage structure of the process definition repository is not part of the WfMC standard. The use of a package is defined only as an aid to simplify the import/export of reusable data structures. Where a simple process repository structure is used, operating at a single level of process definition, shared information within an imported package may be replicated into each of the individual process definitions at the import interface (and similarly repacked, if required, for process definition export).

6.4.1.1. Redefinition and Scope

The possibility of redefining attributes and meta-model entities and referencing external packages introduces the principles of scope and hierarchy into the XPDL (and process repository) structures.

(i) Workflow relevant Data

Workflow process relevant data has a scope that is defined by the directly surrounding meta-model entity and is not nested. The visibility of its identifier is also defined by that entity.

(ii) Attributes

Attributes including extended attributes have a scope that is defined by the directly surrounding meta-model entity and are nested, i.e. may be redefined at a lower level. Example: The name attribute is

redefined in each entity definition. The visibility of extended attribute identifiers is within the particular entity and all sub-entities unless the identifier is redefined in a sub-entity.

(iii) Workflow participants and applications

- Workflow participants and applications have a scope and visibility equivalent to extended attributes. All referenced workflow relevant data and extended attributes have to be defined in the scope where they are used, at least in the same package.

For a referenced external package entity that needs itself reference to entities and their identifiers defined in its external package clause the mechanism is started with the root in that package. That guarantees that no conflict takes place if the invoking process has an entity with the same id, which the definer of the referenced package cannot be aware of.

The described mechanism of external package provides high flexibility for workflow designers and administrators. One can separate organization descriptions (participant entities) and process definitions in separate models, one can add a new release of a process description or add a new process definition sharing the rest of the definition of previously defined and exchanged models without resubmitting the whole context etc.

6.5. Elements Overview

The following table gives an overview of major elements defined within XPDL.

- The first row contains attributes and elements common to all major elements. All major elements have the attributes *id* and *name and may contain a Description and Extended Attributes*.
- The second row contains specific properties of the respective major element.
- The third group consists of elements that may contain references to other elements.
- Documentation and Icon elements contain presentation information to be used by the executing engine.
- The fifth group contains information relevant for simulation and process optimisation (BPR-relevant information).

Further elements and predefined attributes may be added to the model to create future conformance levels. A short description and the semantics of all elements are given in the subsequent chapters.

Package	Workflow Process	Activity	Transition	Application	Data Field (Workflow Relevant Data)	Participant
- Id	- Id	- Id	- Id	- Id	- Id	- Id
- Name	- Name	- Name	- Name	- Name	- Name	- Name
- Description	- Description	- Description	- Description	- Description	- Description	- Description
- Extended Attributes	- Extended Attributes	- Extended Attributes	- Extended Attributes	- Extended Attributes	- Extended Attributes	- Extended Attributes
- XPDL Version	- Creation Date	- Automation Mode			- Data Type	- Participant Type
- Source Vendor ID	- Version	- Split				
- Creation Date	- Author	- Join				
- Version	- Codepage	- Priority				
- Author	- Country Key	- Limit				
- Codepage	- Publication Status	- Start Mode				
- Country Key	- Priority	- Finish Mode				
- Publication Status	- Limit	- Deadline				
- Conformance Class	- Valid From Date					
- Priority Unit	- Valid To Date					

- Responsible - External Package	- Parameters - Responsible	- Performer - Tool - Subflow - ActivitySet - Actual Parameter	- Condition - From - To	- Parameters	- Initial Value	
- Documentation - Icon	- Documentation - Icon	-Documentation- -Icon				
- Cost Unit	-Duration Unit - Duration - Waiting Time - Working Time	-Cost - Duration - Waiting Time - Working Time				

Table 6-1: Overview of Elements

6.5.1. Vendor or User specific Extensions

Although the meta-model and associated XPDL contain most of the constructs, which are likely to be required in the exchange of process definitions, there may be circumstances under which additional information (user or vendor specific) will need to be included within a process definition. Users and vendors are encouraged to work as far as possible within the standard entity / attribute sets; the mechanisms described below to support extension provide a standardized means of expressing the extension for interchange purposes but may require localized system adaptation to provide any associated runtime support during process enactment.

6.5.1.1. Extended Attributes

The primary method to support such extensions is by the use of extended attributes. Extended attributes are those defined by the user or vendor, where necessary, to express any additional entity characteristics which need to be exchanged between systems. Any run-time semantics associated with the use of the extended attribute during process enactment are separately specified and require bilateral agreement between the exporter and the importing workflow service.

6.5.1.2. Extended parameter mapping

No specific details of the scheme for encoding and passing parameter data are defined within this specification. Where parameters are passed on remote subflow invocation using the workflow Interoperability Specification (interface four), specifications are provided for the mapping of such parameters (for example into wf-XML exchanges) using the operations within the concrete syntax specification for interoperability. Any local scheme for parameter mapping and encoding is vendor defined on a product-by-product basis and lies outside the scope of this specification.

7. XML Process Definition Language

7.1. Elements Common for Multiple Entities

7.1.1. Extended Attributes

Extended Attributes can be used in all entities. They allow vendors to extend the functionality of this specification to meet individual product needs. A vendor may add sub-content to an ExtendedAttribute.

```
<xsd:element name="ExtendedAttribute">
    <xsd:complexType mixed="true">
        <xsd:choice minOccurs="0" maxOccurs="unbounded">
            <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
        </xsd:choice>
        <xsd:attribute name="Name" type="xsd:NMTOKEN" use="required"/>
        <xsd:attribute name="Value" type="xsd:string"/>
    </xsd:complexType>
</xsd:element>

<xsd:element name="ExtendedAttributes">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element ref=“xpdl:ExtendedAttribute"
                minOccurs="0" maxOccurs="unbounded"/>
        </xsd:sequence>
    </xsd:complexType>
</xsd:element>
```

	Description
Name	Used to identify the Extended Attribute
Value	Value required for a particular product.

Table 7-1: Extended Attributes -- Attributes

7.1.2. Formal Parameters

Formal parameters can be used as attributes in workflow process and workflow application. They are passed during invocation and return of control (e.g. of an invoked application). These are the invocation parameters.

```
<xsd:element name="FormalParameter">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element ref=“xpdl:DataType"/>
            <xsd:element ref=“xpdl:Description" minOccurs="0"/>
        </xsd:sequence>
        <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
        <xsd:attribute name="Index" type="xsd:NMTOKEN"/>
        <xsd:attribute name="Mode" default="IN">
            <xsd:simpleType>
                <xsd:restriction base="xsd:NMTOKEN">
                    <xsd:enumeration value="IN"/>
                    <xsd:enumeration value="OUT"/>
                    <xsd:enumeration value="INOUT"/>
                </xsd:restriction>
            </xsd:simpleType>
```



```
        </xsd:attribute>
    </xsd:complexType>
</xsd:element>

<xsd:element name="FormalParameters">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element ref="xpdl:FormalParameter"
                minOccurs="0" maxOccurs="unbounded"/>
        </xsd:sequence>
    </xsd:complexType>
</xsd:element>
```

	Description	
Data type	Data type of the formal parameter. See Section 7.9	
Description	Textual description of the formal parameter	
Id	Identifier for the parameter	
Index	Index of the parameter	
Mode	IN	Input Parameters
	OUT	Output Parameters
	INOUT	Parameters used as input and output

Table 7-3: Formal Parameters – Attributes

7.1.2.1. Parameter passing semantics

The parameter passing semantics is defined as:

(a) Any read-only formal parameters (IN) are initialised by the value of the corresponding actual parameter in the call (an expression). This is pass-by-value semantics.

(b) Any read/write formal parameters (INOUT) are initialised by the value of the corresponding actual (passed) parameter, which must be the identifier of a workflow relevant data entity. On completion of the process, the value of the formal out parameter is copied back to the original actual parameter (which must be the identifier of a workflow relevant data entity). This is copy-restore semantics.

(c) Any write-only formal parameters (OUT) are initialised to zero (strings will be set to the empty string, complex data will have each element set to zero). On completion of the process, the value of the formal out parameter is copied back to the original actual parameter (which must be the identifier of a workflow relevant data entity). This is zero-restore semantics.

7.1.2.2. Concurrency semantics

Copying and restoring of parameters are treated as atomic operations; to avoid access conflicts from concurrent operations on workflow relevant data within the process instance these operations are serialized. Between copy and restore of (c) no locking is assumed and the returned parameter value will overwrite the local value (of the particular workflow relevant data item) at the time of the return call.

7.1.2.3. Formal-actual parameter mapping

The mapping of actual to formal parameters during invocation is defined by a parameter map list. The actual parameters are mapped 1:1 to the formal parameters in sequence, i.e. the first actual maps to the first formal, the second actual maps to the second formal etc. Type compatibility is required within the definitions and may be enforced by the run-time workflow system. The effects of violation are locally defined and do not form part of this specification

In case the actual parameter is an expression, the expression is evaluated and buffered by the Workflow engine, and the contents of this buffer is used for formal-actual mapping. How the buffering and mapping is performed is outside the scope if this document.

7.1.3. External Reference

ExternalReference is a reference to an external definition of an entity. It can be used in DataTypes, Participant, and Application.

```
<xsd:element name="ExternalReference">
   <xsd:complexType>
      <xsd:attribute name="xref" type="xsd:NMTOKEN" use="optional"/>
      <xsd:attribute name="location" type="xsd:anyURI" use="required"/>
      <xsd:attribute name="namespace" type="xsd:anyURI" use="optional"/>
   </xsd:complexType>
</xsd:element>
```

	Description
Location	It specifies the URI of the document that defines the type.
Namespace	It allows specification of the scope in which the entity is defined.
xref	It specifies the identity of the entity within the external document.

Table 7-5: External Reference -- Attributes

Example 1: A FormalParameter that is defined by an XML schema:

```
<FormalParameter Id="PO">
   <DataType>
      <ExternalReference location="http://abc.com/schemas/po.xsd"/>
   </DataType>
   <Description>PO specification for abc.com</Description>
</FormalParameter>
```

Example 2: A DataField defined by a Java class:

```
<DataField Id="PO" Name="PurchaseOrder" IsArray="FALSE">
   <DataType>
      <ExternalReference location="com.abc.purchases.PO"/>
   </DataType>
   <Description>PO specification for abc.com</Description>
</DataField>
```

7.1.3.1. Web Services

An activity in a process may invoke a web service. The ExternalReference element may be used as a reference to applications and data types that are defined in Web Service (WSDL) documents.

Example 3: A DataField whose data type is defined in a WSDL document:

```
<DataField Id="abcPO" Name=”abcPurchaseOrder” IsArray=”False”>
   <DataType>
      <ExternalReference xref="PO"
location="http://abc.com/services/poService.wsdl"
namespace="poService/definitions/types"/>
   </DataType>
</DataField>
```

Example 4: An Application that is defined as an operation in a WSDL document:

```
<Application Id="placeOrder">
   <ExternalReference location="http://abc.com/PO/services/poService.wsdl"
      xref="PlaceOrder" namespace=
         "http://abc.com/services/poService.wsdl/definitions/portType"/>
```



```
</Application>
```

7.2. Package Definition

It is possible to define several processes within one package, which may share the same tools and participants. We recommend creating one package per business process which should contain all the necessary workflow processes as well as all the associated tools and workflow participants, although it is not required. Also it is possible to define just parts of one process definition or common parts of several processes within one package (e.g. a workflow participant list or a workflow application list).

```
<xsd:element name="Package">
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element ref="xpdl:PackageHeader"/>
         <xsd:element ref="xpdl:RedefinableHeader" minOccurs="0"/>
         <xsd:element ref="xpdl:ConformanceClass" minOccurs="0"/>
         <xsd:element ref="xpdl:Script" minOccurs="0"/>
         <xsd:element ref="xpdl:ExternalPackages" minOccurs="0"/>
         <xsd:element ref="xpdl:TypeDeclarations" minOccurs="0"/>
         <xsd:element ref="xpdl:Participants" minOccurs="0"/>
         <xsd:element ref="xpdl:Applications" minOccurs="0"/>
         <xsd:element ref="xpdl:DataFields" minOccurs="0"/>
         <xsd:element ref="xpdl:WorkflowProcesses" minOccurs="0"/>
         <xsd:element ref="xpdl:ExtendedAttributes" minOccurs="0"/>
      </xsd:sequence>
      <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
      <xsd:attribute name="Name" type="xsd:string"/>
   </xsd:complexType>
</xsd:element>
```

	Description
Applications	A list of Workflow Application Declarations. See section 7.3
Conformance Class	Structural restriction on process definitions in this package. See section 7.2.3
Data Fields	A list of Workflow Relevant Data defined for the package. See section 7.8
Extended Attributes	A list of vendor-defined extensions that may be added to the package. See section 7.1.1
External Packages	Reference to another Package definition defined in a separate document.
Id	Used to identify the package.
Name	Text. Used to identify the package.
Package Header	A set of elements specifying package characteristics.
Participants	A list of resources used in implementing processes in the package. See section 7.7
Redefinable Header	A set of elements and attributes used by both the Package and Process definitions.
Script	Identifies the scripting language used in expressions.
Type Declarations	A list of Data Types used in the package. See section 7.9
Workflow Processes	A list of the Workflow Processes that comprise this package. See section 7.4

Table 7-7: Package Definition -- Attributes

7.2.1. Package definition Header

The package definition header keeps all information central to a package such as XPDL version, source vendor id, etc.


```
<xsd:element name="PackageHeader">
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element ref=“xpdl:XPDLVersion"/>
         <xsd:element ref=“xpdl:Vendor"/>
         <xsd:element ref=“xpdl:Created"/>
         <xsd:element ref=“xpdl:Description" minOccurs="0"/>
         <xsd:element ref=“xpdl:Documentation" minOccurs="0"/>
         <xsd:element ref=“xpdl:PriorityUnit" minOccurs="0"/>
         <xsd:element ref=“xpdl:CostUnit" minOccurs="0"/>
      </xsd:sequence>
   </xsd:complexType>
</xsd:element>

<xsd:element name="XPDLVersion" type="xsd:string"/>

<xsd:element name="Vendor" type="xsd:string"/>

<xsd:element name="Created" type="xsd:string"/>

<xsd:element name="Description" type="xsd:string"/>

<xsd:element name="Documentation" type="xsd:string"/>

<xsd:element name="PriorityUnit" type="xsd:string"/>

<xsd:element name="CostUnit" type="xsd:string"/>
```

	Description
Cost Unit	Units used in Simulation Data (Usually expressed in terms of a currency)
Created	Creation date of Package Definition.
Description	Textual description of the package
Documentation	Operating System specific path- and filename of help file/description file.
Priority Unit	A text string with user defined semantics.
Vendor	Defines the origin of this model definition and contains vendor's name, vendor's product name and product's release number.
XPDL Version	Version of this specification. The current value, for this specification, is “0.02”.

Table 7-9: Package Definition Header – Attributes

7.2.2. Redefinable Header

The redefinable header covers those header attributes that may be defined in the workflow definition header and may be redefined in the header of any process definition. In case of redefinition, the scope rules hold.

```
<xsd:element name="RedefinableHeader">
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element ref=“xpdl:Author" minOccurs="0"/>
         <xsd:element ref=“xpdl:Version" minOccurs="0"/>
         <xsd:element ref=“xpdl:Codepage" minOccurs="0"/>
         <xsd:element ref=“xpdl:Countrykey" minOccurs="0"/>
```

```
                <xsd:element ref="xpdl:Responsibles" minOccurs="0"/>
            </xsd:sequence>
            <xsd:attribute name="PublicationStatus">
                <xsd:simpleType>
                    <xsd:restriction base="xsd:NMTOKEN">
                        <xsd:enumeration value="UNDER_REVISION"/>
                        <xsd:enumeration value="RELEASED"/>
                        <xsd:enumeration value="UNDER_TEST"/>
                    </xsd:restriction>
                </xsd:simpleType>
            </xsd:attribute>
        </xsd:complexType>
    </xsd:element>

    <xsd:element name="Author" type="xsd:string"/>

    <xsd:element name="Version" type="xsd:string"/>

    <xsd:element name="Codepage" type="xsd:string"/>

    <xsd:element name="Countrykey" type="xsd:string"/>

    <xsd:element name="Responsible" type="xsd:string"/>

    <xsd:element name="Responsibles">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element ref="xpdl:Responsible" minOccurs="0"
maxOccurs="unbounded"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
```

	Description
Author	Name of the author of this package definition.
Code page	The codepage used for the text parts
Country key	Country code based on ISO 3166. It could be either the three digits country code number, or the two alpha characters country codes.
Publication Status	Status of the Workflow Process Definition. UNDER_REVISION RELEASED UNDER_TEST
Responsible(s)	Workflow participant, who is responsible for this workflow process; the supervisor during run time Link to entity workflow participant. Workflow participant, who is responsible for this workflow of this Model definition (usually an Organisational Unit or a Human). It is assumed that the responsible is the supervisor during run time. Default: Initiating participant.
Version	Version of this Package Definition.

Table 7-11: *Redefinable Header – Attributes*

7.2.3. Conformance Class Declaration

The conformance class declaration allows description of the conformance class to which the definitions in this model definition are restricted. The specified class applies to all the contained process definitions, unless it is re-defined locally at the process definition level.

```
<xsd:element name="ConformanceClass">
   <xsd:complexType>
      <xsd:attribute name="GraphConformance">
         <xsd:simpleType>
            <xsd:restriction base="xsd:NMTOKEN">
               <xsd:enumeration value="FULL_BLOCKED"/>
               <xsd:enumeration value="LOOP_BLOCKED"/>
               <xsd:enumeration value="NON_BLOCKED"/>
            </xsd:restriction>
         </xsd:simpleType>
      </xsd:attribute>
   </xsd:complexType>
</xsd:element>
```

	Description	
Conformance Class	FULL-BLOCKED	The network structure is restricted to proper nesting of SPLIT/JOIN and loops.
	LOOP-BLOCKED	The network structure is restricted to proper nesting of loops.
	NON-BLOCKED	There is no restriction on the network structure. This is the default.

Table 7-13: Conformance Class Declaration – Attributes

7.2.4. Script

The Script element identifies the scripting language used in XPDL expressions. A text expression may be used wherever an element is of type xsd:string. One could, for example, use an expression within the ActualParameter or Cost elements.

An expression composed of formatted XML (e.g., MathML) may be used within the Xpression element (used within a Transition Condition).

```
<xsd:element name="Script">
   <xsd:complexType>
      <xsd:attribute name="Type" type="xsd:string" use="required"/>
      <xsd:attribute name="Version" type="xsd:string" use="optional"/>
      <xsd:attribute name="Grammar" type="xsd:anyURI" use="optional"/>
   </xsd:complexType>
</xsd:element>
```

	Description
Type	Identifies the scripting language used in expressions. For consistency across implementations, when specifying a standard scripting languages, it is recommended that the Type be selected from the following strings: text/javascript, text/vbscript, text/tcl, text/ecmascript, text/xml.
Version	This is the version of the scripting language.
Grammar	This is a reference to a document that specifies the grammar of the language. It could be, for example, an XML schema, a DTD, or a BNF.

Table 7-15: Script – Attributes

7.2.5. External Package Reference

External package reference allows referencing definitions in another Package definition or in other systems providing an Interface to the Workflow Management system (e.g. a legacy Organisation Description Management Tool).

```
<xsd:element name="ExternalPackage">
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element ref=“xpdl:ExtendedAttributes" minOccurs="0"/>
      </xsd:sequence>
      <xsd:attribute name="href" type="xsd:string"/>
   </xsd:complexType>
</xsd:element>

<xsd:element name="ExternalPackages">
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element ref=“xpdl:ExternalPackage"
            minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
   </xsd:complexType>
</xsd:element>
```

	Description
Extended Attributes	Optional vendor-defined extensions to meet implementation needs. See section 7.1.1
href	A Model Identifier. Logical reference to a Model

Table 7-16: External Package Reference -- Attributes

7.3. Workflow Application Declaration

Workflow application declaration is a list of all applications or tools required and invoked by the workflow processes defined within the process definition or surrounding package. Tools may be defined (or, in fact, just named). This means, that the real definition of the tools is not necessary and may be handled by an object manager. The reason for this approach is the handling of multi-platform environments, where a different program (or function) has to be invoked for each platform. XPDL abstracts from the concrete implementation or environment (thus these aspects are not of interest at process definition time).

```
<xsd:element name="Application">
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element ref=“xpdl:Description" minOccurs="0"/>
         <xsd:choice>
            <xsd:element ref=“xpdl:FormalParameters"/>
            <xsd:element ref=“xpdl:ExternalReference" minOccurs="0"/>
         </xsd:choice>
         <xsd:element ref=“xpdl:ExtendedAttributes" minOccurs="0"/>
      </xsd:sequence>
      <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
      <xsd:attribute name="Name" type="xsd:string"/>
   </xsd:complexType>
</xsd:element>

<xsd:element name="Applications">
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element ref=“xpdl:Application" minOccurs="0"
```

```
maxOccurs="unbounded"/>
          </xsd:sequence>
       </xsd:complexType>
    </xsd:element>
```

	Description
Description	Short textual description of the application.
Extended Attributes	Optional vendor-defined extensions to meet implementation needs. See section 7.1.1
External Reference	A reference to an external specification of the application signature. See section 7.1.3
Formal Parameters	A list of parameters that is interchanged with the application via the invocation interface. See section 7.1.2.
Id	Used to identify the workflow application definition
Name	Text used to identify an application (may be interpreted as a generic name of the tool).

Table 7-18: Workflow Application Declaration -- Attributes

7.3.1. Invocation Parameters

A Workflow Application declaration may have parameter definitions for the (invocation) parameters and also use them within other entities.

Copying the invocation `IN` is treated as one atomic operation. The same holds for restoring the invocation `OUT`. Between these two operations no assumption is made about concurrency behaviour.

7.4. Workflow Process Definition

The Workflow Process Definition defines the elements that make up a workflow. It contains definitions or declarations, respectively, for Activity and, optionally, for Transition, Application, and Process Relevant Data entities. Attributes may be specified for administration relevant data like author, and version; for runtime relevant data like priority; and for BPR and simulation relevant data.

A Workflow Process may run as an implementation of an activity of type subflow; in this case parameters may be defined as attributes of the process.

Where a workflow process definition includes input parameters and is instantiated by means other than a subflow call (for example by local event) the method for initializing any input parameters is locally defined. In such circumstances any workflow relevant data associated with the instantiated process definition, which is included within the parameter list will be initialized to the value specified in the “default value” (where specified). Where workflow relevant data is not passed as an input parameter, or initialized by “default value” the result is undefined. Similarly where a subflow terminates abnormally without returning out parameter values to the calling process, the result is undefined.

In general the scope of the defined entity identifier and name is the surrounding entity. The identifier is unique in this scope. For the Process identifier and name the scope is the surrounding Package.

```
<xsd:element name="WorkflowProcess">
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element ref=“xpdl:ProcessHeader"/>
         <xsd:element ref=“xpdl:RedefinableHeader" minOccurs="0"/>
         <xsd:element ref=“xpdl:FormalParameters" minOccurs="0"/>
         <xsd:group ref="xpdl:DataTypes"/>
         <xsd:element ref=“xpdl:DataFields" minOccurs="0"/>
         <xsd:element ref=“xpdl:Participants" minOccurs="0"/>
         <xsd:element ref=“xpdl:Applications" minOccurs="0"/>
         <xsd:element ref="xpdl:ActivitySets" minOccurs="0"/>
         <xsd:element ref=“xpdl:Activities" minOccurs="0"/>
```



```
            <xsd:element ref="xpdl:Transitions" minOccurs="0"/>
            <xsd:element ref="xpdl:ExtendedAttributes" minOccurs="0"/>
        </xsd:sequence>
        <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
        <xsd:attribute name="Name" type="xsd:string"/>
        <xsd:attribute name="AccessLevel">
            <xsd:simpleType>
                <xsd:restriction base="xsd:NMTOKEN">
                    <xsd:enumeration value="PUBLIC"/>
                    <xsd:enumeration value="PRIVATE"/>
                </xsd:restriction>
            </xsd:simpleType>
        </xsd:attribute>
    </xsd:complexType>
</xsd:element>

<xsd:element name="WorkflowProcesses">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element ref="xpdl:WorkflowProcess"
                minOccurs="0" maxOccurs="unbounded"/>
        </xsd:sequence>
    </xsd:complexType>
</xsd:element>
```

	Description
AccessLevel	The Access level of a process may be either PUBLIC or PRIVATE. If PUBLIC the process may be invoked by an external system or application. A process with private access may only be invoked from a SubFlow Activity (see Section 7.5.4.3.
Activities	A list of activities that comprise the process. See section 7.4.3.
ActivitySets	A list of self contained sets of activities and transitions.
Applications	A list of Workflow Application Declarations. See section 7.3.
Data Fields	A list of Workflow Relevant Data defined for the process. See section 7.8.
Extended Attributes	Optional vendor-defined extensions to meet implementation needs. See section 7.1.1
Formal Parameters	A list of parameters that may be passed to the process. See section 7.1.2.
Id	Used to identify the workflow process.
Name	Text Used to identify the workflow process.
Participants	A list of resources used in implementing the process. See section 7.7.
Process Header	A set of elements specifying process characteristics.
Redefinable Header	A set of elements and attributes used by both the Package and Process definitions.
Transitions	A list of the transitions that connect the process activities. See section 7.6.

Table 7-20: Workflow Process Definition -- Attributes

7.4.1. Workflow Process Definition Header

The workflow process definition header keeps all information specific for a process definition such as process version, priority, duration of validity, etc.

```
<xsd:element name="ProcessHeader">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element ref="xpdl:Created" minOccurs="0"/>
```

```
            <xsd:element ref="xpdl:Description" minOccurs="0"/>
            <xsd:element ref="xpdl:Priority" minOccurs="0"/>
            <xsd:element ref="xpdl:Limit" minOccurs="0"/>
            <xsd:element ref="xpdl:ValidFrom" minOccurs="0"/>
            <xsd:element ref="xpdl:ValidTo" minOccurs="0"/>
            <xsd:element ref="xpdl:TimeEstimation" minOccurs="0"/>
         </xsd:sequence>
         <xsd:attribute name="DurationUnit">
            <xsd:simpleType>
               <xsd:restriction base="xsd:NMTOKEN">
                  <xsd:enumeration value="Y"/>
                  <xsd:enumeration value="M"/>
                  <xsd:enumeration value="D"/>
                  <xsd:enumeration value="h"/>
                  <xsd:enumeration value="m"/>
                  <xsd:enumeration value="s"/>
               </xsd:restriction>
            </xsd:simpleType>
         </xsd:attribute>
      </xsd:complexType>
   </xsd:element>

   <xsd:element name="Created" type="xsd:string"/>

   <xsd:element name="Description" type="xsd:string"/>

   <xsd:element name="Limit" type="xsd:string"/>

   <xsd:element name="Priority" type="xsd:string"/>

   <xsd:element name="TimeEstimation">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:WaitingTime" minOccurs="0"/>
            <xsd:element ref="xpdl:WorkingTime" minOccurs="0"/>
            <xsd:element ref="xpdl:Duration" minOccurs="0"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>

   <xsd:element name="WaitingTime" type="xsd:string"/>

   <xsd:element name="WorkingTime" type="xsd:string"/>

   <xsd:element name="Duration" type="xsd:string"/>

   <xsd:element name="ValidFrom" type="xsd:string"/>

   <xsd:element name="ValidTo" type="xsd:string"/>
```

	Description
Created	Creation date of workflow process definition.
Description	Short textual description of the process.
Duration	Expected duration time to perform a task in units of DurationUnit.

	Description
Duration Unit	Describes the default unit to be applied to an integer duration value that has no unit tag. Possible units are: Y - year M - month D - day H - hour m - minute s - second
Limit	Expected duration for time management purposes (e.g. starting an escalation procedure etc.) in units of DurationUnit. It is counted from the starting date/time of the Process. The consequences of reaching the limit value are not defined in this document (i.e. vendor specific). It is assumed that in this case at least the Responsible of the current process is notified of this situation.
Priority	The priority of the process type. Default: Inherited from Model Definition.
Time Estimation	Grouping of waiting time, working time, and duration. Used for simulation purposes.
Valid From	The date that the workflow process definition is active from. Empty string means system date. Default: Inherited from Model Definition.
Valid To	The date at which the process definition becomes valid. Empty string means unlimited validity. Default: Inherited from Model Definition.
Waiting Time	Describes the amount of time, which is needed to prepare the performance of the task (time estimation) (waiting time is provided by the analysis environment and may be updated by the runtime environment) in units of DurationUnit.
Working Time	Describes the amount of time the performer of the activity needs to perform the task (time estimation) (working time is needed for analysis purposes and is provided by the evaluation of runtime parameters) in units of DurationUnit.

Table 7-22: Workflow Process Definition Header -- Attributes

7.4.2. Workflow Process Redefinable Header

```
<xsd:element name="RedefinableHeader">
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element ref=“xpdl:Author" minOccurs="0"/>
         <xsd:element ref=“xpdl:Version" minOccurs="0"/>
         <xsd:element ref=“xpdl:Codepage" minOccurs="0"/>
         <xsd:element ref=“xpdl:Countrykey" minOccurs="0"/>
         <xsd:element ref=“xpdl:Responsibles" minOccurs="0"/>
      </xsd:sequence>
      <xsd:attribute name="PublicationStatus">
         <xsd:simpleType>
            <xsd:restriction base="xsd:NMTOKEN">
               <xsd:enumeration value="UNDER_REVISION"/>
               <xsd:enumeration value="RELEASED"/>
               <xsd:enumeration value="UNDER_TEST"/>
            </xsd:restriction>
         </xsd:simpleType>
```

```
            </xsd:attribute>
         </xsd:complexType>
      </xsd:element>

      <xsd:element name="Author" type="xsd:string"/>

      <xsd:element name="Codepage" type="xsd:string"/>

      <xsd:element name="Countrykey" type="xsd:string"/>

      <xsd:element name="Responsible" type="xsd:string"/>

      <xsd:element name="Responsibles">
         <xsd:complexType>
            <xsd:sequence>
               <xsd:element ref=“xpdl:Responsible" minOccurs="0"
maxOccurs="unbounded"/>
            </xsd:sequence>
         </xsd:complexType>
      </xsd:element>

      <xsd:element name="Version" type="xsd:string"/>
```

	Description
Author	Name of the author of this workflow process definition. (The one, who put it into the repository)
Codepage	The codepage used for the text parts. Default: Inherited from Model Definition.
Country key	Country code based on ISO 3166. It could be either the three digits country code number, or the two alpha characters country codes. Default: Inherited from Model Definition.
Publication Status	Status of the Workflow Process Definition. Default: Inherited from Model Definition. UNDER_REVISION RELEASED UNDER_TEST
Responsible(s)	Workflow participant, who is responsible for this workflow process (usually an Organisational Unit or a Human). It is assumed that the responsible is the supervisor during execution of the process. Default: Inherited from Model Definition.
Version	Version of this workflow process definition.

Table 7-24: Workflow Process Redefinable Header – Attributes

7.4.3. Activity Set

An activity set is a self-contained set of activities and transitions. Transitions in the set should refer only to activities in the same set and there should be no transitions into or out of the set. Activity sets can be executed by block activities (see Section 7.5.2).

```
      <xsd:element name="ActivitySet">
         <xsd:complexType>
            <xsd:sequence>
               <xsd:element ref="xpdl:Activities" minOccurs="0"/>
               <xsd:element ref="xpdl:Transitions" minOccurs="0"/>
            </xsd:sequence>
            <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
```

```
            </xsd:complexType>
        </xsd:element>

        <xsd:element name="ActivitySets">
            <xsd:complexType>
                <xsd:sequence>
                    <xsd:element ref="xpdl:ActivitySet" minOccurs="0"
    maxOccurs="unbounded"/>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
```

	Description
Activities	A list of activities that comprise the process. See section 7.4.3.
Id	Used to identify the workflow process.
Transitions	A list of the transitions that connect the process activities. See section 7.6.

Table 7-26: ActivitySet

7.5. Workflow Process Activity

The Workflow Activity Definition is used to define each elementary activity that makes up a workflow process. Attributes may be defined to specify activity control information, implementation alternatives, performer assignment, runtime relevant information like priority, and data used specifically in BPR and simulation situations (and not used within workflow enactment). In addition, restrictions on data access and to transition evaluation (e.g. Split and Join) can be described. Mandatory attributes are used to define the activity identifier and type; a small number of other attributes are optional but have common usage across all activity types. Other attribute usage depends upon the activity type as shown in the table below.

For the Activity identifier and name the scope is the surrounding workflow process.

The activity description is used to describe several different activity types. All these activities share the same (common) general activity attributes, but the usage of other attributes, particularly participant and application assignment and the use of workflow relevant data may be specialized to the activity type. The following table identifies the usage of other attributes / entity types for the different activity types.

Entity Types (usage within Activity Type)	Activity Type				
	Implementation Type			Route	BlockActivity
	None	Application	Subflow		
Transition Restriction	Normal	Normal	Normal, plus subflow call / return within activity	Normal; any additional controls implemented within Route activity	Normal; refers to activities within same context, not to activites within ActivitySet
Participant Assignment	Normal	Normal	N/A	N/A	N/A
Application Assignment	None	Yes	N/A	N/A	N/A
Use of workflow Relevant Data	Normal	Normal	May be used in parameter passing	May be used in routing control conditions	May be used in routing control conditions

Table 7-27: Entity type relationships for different Activity types

Notes on usage:

Transition restrictions, subflow, and route activities are described in the section on transitions. In general, normal transition restrictions may be declared at the level of the activity boundary within the surrounding process, whereas specialized flow conditions (subflow, or the internal part of a route activity) operate "internal" to the activity (but may reference activities within the surrounding process definition). The following diagram illustrates the generic structure of an activity and the above variants.

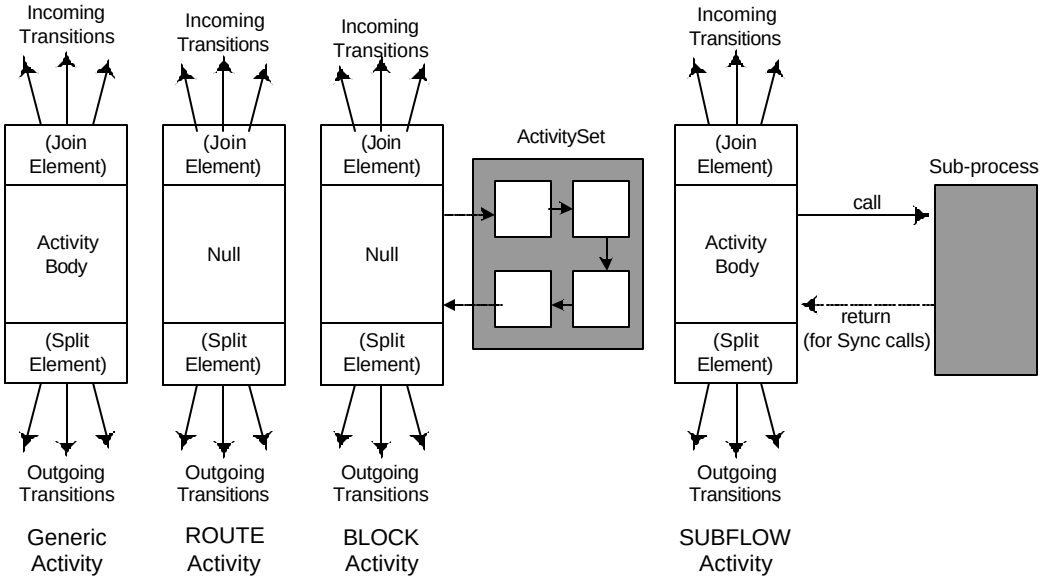

Figure 7-1: Activity Structures & Transition Conditions

Where the implementation type is NONE, the workflow activity is manually controlled and its completion must be explicitly signaled to the workflow management system. Such activities might typically comprise instructions to the participant to undertake a non-automated task of some type and inform a supervisor when completed.

Workflow relevant data may (potentially) be referenced within any activity although its use in manual activities is undefined through the process definition. Where an activity is of type subflow any in-parameters passed to the called (sub-) process must have been declared as workflow relevant data within the calling process / activity definition, or have been inherited from the surrounding package. (Similar requirements apply to any out-parameters returned to the calling process.) Routing and block activities do not manipulate workflow relevant data directly, but may refer to such data within conditional expressions within the join/split control logic.

```
    <xsd:element name="Activity">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element ref="xpdl:Description" minOccurs="0"/>
                <xsd:element ref="xpdl:Limit" minOccurs="0"/>
                <xsd:choice>
                    <xsd:element ref="xpdl:Route"/>
                    <xsd:element ref="xpdl:Implementation"/>
                    <xsd:element ref="xpdl:BlockActivity"/>
                </xsd:choice>
                <xsd:element ref="xpdl:Performer" minOccurs="0"/>
                <xsd:element ref="xpdl:StartMode" minOccurs="0"/>
                <xsd:element ref="xpdl:FinishMode" minOccurs="0"/>
                <xsd:element ref="xpdl:Priority" minOccurs="0"/>
                <xsd:element ref="xpdl:Deadline" minOccurs="0"
maxOccurs="unbounded"/>
                <xsd:element ref="xpdl:SimulationInformation" minOccurs="0"/>
                <xsd:element ref="xpdl:Icon" minOccurs="0"/>
                <xsd:element ref="xpdl:Documentation" minOccurs="0"/>
```

```
                <xsd:element ref="xpdl:TransitionRestrictions" minOccurs="0"/>
                <xsd:element ref="xpdl:ExtendedAttributes" minOccurs="0"/>
            </xsd:sequence>
            <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
            <xsd:attribute name="Name" type="xsd:string"/>
        </xsd:complexType>
    </xsd:element>

    <xsd:element name="Activities">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element ref="xpdl:Activity" minOccurs="0"
maxOccurs="unbounded"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>

    <xsd:element name="Performer" type="xsd:string"/>

    <xsd:element name="Icon" type="xsd:string"/>
```

	Description
BlockActivity	An Activity that executes an ActivitySet.
Deadline	Specification of a deadline and action to be taken if it is reached.
Description	Textual description of the activity.
Documentation	The address (e.g. path- and filename) for a help file or a description file of the activity.
Extended Attributes	Optional extensions to meet individual implementation needs
Finish Mode	Describes how the system operates at the end of the Activity.
Icon	Address (path- and filename) for an icon to represent the activity.
Id	Used to identify the workflow process activity.
Implementation	A "regular" Activity. Mandatory if not a Route. Alternative implementations are “no”, or “subflow”
Limit	Expected duration for time management purposes (e.g. starting an escalation procedure etc.) in units of DurationUnit. It is counted from the starting date/time of the Process. The consequences of reaching the limit value are not defined in this document (i.e. vendor specific).
Name	Text Used to identify the workflow process activity.
Performer	Link to entity workflow participant. May be an expression. Default: Any Participant.
Priority	A value that describes the initial priority of this activity when it starts execution. If this attribute is not defined but a priority is defined in the Process definition then that is used. By default it is assumed that the priority levels are the natural numbers starting with zero, and that the higher the value the higher the priority (i.e.: 0, 1,…, n).
Route	A "dummy" Activity
Simulation Information	Estimations for simulation of an Activity. No default.
Start Mode	Describes how the execution of an Activity is triggered.
Transition Restrictions	Provides further restrictions and context-related semantics description of Transitions

Table 7-29: Process Activity -- Attributes

7.5.1. Route Activity

The Route Activity is a "dummy" Activity that permits the expression of "cascading" Transition conditions (e.g. of the type "`IF condition-1 THEN TO Activity-1 ELSE IF condition-2 THEN TO Activity-2 ELSE Activity-3 ENDIF`"). Some vendors might implement "cascading" transition conditions directly without requiring an activity counterpart for a route; others might require it. Wherever possible vendors and process designers are encouraged to structure such cascading conditions as an XOR split from the outgoing activity. Certain transition combinations cannot be expressed within a single transition list from the outgoing activity or a single incoming list to an activity. These cases require the use of one or more dummy activities; examples are:

- Combination of XOR and AND split conditions on outgoing transitions from an activity.
- Combination of XOR and AND join conditions on incoming transitions to an activity
- Transitions involving conditional AND joins of a subset of threads, with continuation of individual threads

A route activity has neither a performer nor an application and its execution has no effect on workflow relevant data or application data.

For simulation purposes the following simulation data values should be assumed: `Duration` 0, `Cost` "0", `WorkingTime` 0, `WaitingTime` 0. For `Priority` and `Instantiation` the maximum value should be assumed.

```
<xsd:element name="Route">
   <xsd:complexType/>
</xsd:element>
```

7.5.2. Block Activity

A block activity executes an ActivitySet or self-contained activities/transitions map. From the Block Activity execution proceeds to the first activity in the set and continues within the set until it reaches an exit activity (an activity with no output transitions). Execution then returns to follow the output transitions of the block activity.

7.5.3. Execution Control Attributes

These are attributes of an Activity that allow the definition of various activity-specific features for Activity execution control.

Automation mode defines the degree of automation when triggering and terminating an activity. There are two automation modes:

- ***Automatic mode*** is fully controlled by the workflow engine, i.e. the engine proceeds with execution of the activity within the workflow automatically, as soon as any incoming transition conditions are satisfied. Similarly, completion of the activity and progression to any post activity conditional logic occurs automatically on termination of the final invoked application.
- ***Manual mode*** requires explicit user interaction to cause activity start or finish. In such systems the activity start and/or completion is as a result of explicit user action.

The automation modes can be specified independently for the ***start*** and ***end*** of an Activity.

```
<xsd:element name="StartMode">
   <xsd:complexType>
      <xsd:choice>
         <xsd:element ref="xpdl:Automatic"/>
         <xsd:element ref="xpdl:Manual"/>
      </xsd:choice>
   </xsd:complexType>
</xsd:element>

<xsd:element name="FinishMode">
   <xsd:complexType>
      <xsd:choice>
         <xsd:element ref="xpdl:Automatic"/>
         <xsd:element ref="xpdl:Manual"/>
```

```
            </xsd:choice>
        </xsd:complexType>
    </xsd:element>

    <xsd:element name="Automatic">
        <xsd:complexType/>
    </xsd:element>

    <xsd:element name="Manual">
        <xsd:complexType/>
    </xsd:element>
```

	Description
Start Mode	Describes how the execution of an activity is triggered.
Automatic	Triggered implicitly by the system. Default.
Manual	Triggered explicitly by the end user.
Finish Mode	Describes how the system operates at the end of the activity.
Automatic	Implies an automatic return when the invoked application finishes control. Default.
Manual	The end user has to terminate the activity explicitly.

Table 7-31: Execution Control -- Attributes

7.5.4. Impleme ntation Alternatives

It is assumed that the execution of the Activity is atomic with respect to the data under control of the Workflow engine. That implies that in the case of a system crash, an abort, or a cancellation of the Activity, the Workflow Relevant Data and the workflow control data are rolled back (automatically or by other means), or an appropriate compensating activity is applied. (This does not necessarily hold for audit data.) This version of the specification does not include any specific controls over data synchronization or recovery (for example between workflow execution, subflows or applications under execution.

```
<xsd:element name="Implementation">
    <xsd:complexType>
        <xsd:choice>
            <xsd:element ref=“xpdl:No"/>
            <xsd:element ref=“xpdl:Tool" maxOccurs="unbounded"/>
            <xsd:element ref=“xpdl:SubFlow"/>
        </xsd:choice>
    </xsd:complexType>
</xsd:element>
```

An Activity may be implemented in one of four ways as described in the following table:

	Description
No implementation	Implementation by manual procedures (i.e. not supported by workflow)
Tool	Implementation is supported by (one or more) application(s)
Subflow	Implementation by another process

Table 7-33: Implementation Alternatives -- Attributes

7.5.4.1. No Implementation

No Implementation means that the implementation of this Activity is not supported by Workflow using automatically invoked applications or procedures. Two Alternatives have been identified as to how this may be used:

It is a Manual Activity. In this case FinishMode value Manual is required.

It is an "implicit" activity, which is known to the Workflow Engine (e.g. by vendor-specific Extended Attributes) in terms of any processing requirements. An example is the Pre- and Post-processing Activities in a Workflow, which generate and clear hidden data when starting and terminating a process (e.g. managing the relationship to imaging system and archive). In this case the StartMode and FinishMode values Automatic are common.

(Note that application initiation may still be handled directly by the participant under local control in a manual activity; this lies outside the scope of the specification.)

```
<xsd:element name="No">
    <xsd:complexType/>
</xsd:element>
```

7.5.4.2. Tool

The Activity is implemented by (one or more) tools. A tool may be an application program (link to entity Workflow Application); which may be invoked via IF3 - see the Workflow Client Application API (WAPI - Interface 2).

```
<xsd:element name="Tool">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element ref="xpdl:ActualParameters" minOccurs="0"/>
            <xsd:element ref="xpdl:Description" minOccurs="0"/>
            <xsd:element ref="xpdl:ExtendedAttributes" minOccurs="0"/>
        </xsd:sequence>
        <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
        <xsd:attribute name="Type">
            <xsd:simpleType>
                <xsd:restriction base="xsd:NMTOKEN">
                    <xsd:enumeration value="APPLICATION"/>
                    <xsd:enumeration value="PROCEDURE"/>
                </xsd:restriction>
            </xsd:simpleType>
        </xsd:attribute>
    </xsd:complexType>
</xsd:element>
```

	Description	
Actual Parameters	A list of parameters to be passed to the subflow. See section 7.1.2.3.	
Description	Textual description	
Extended Attributes	Optional extensions to meet individual implementation needs	
Id	Identifier used to identify the application or procedure, depending on the Type.	
Type	APPLICATION	A tool identifier
	PROCEDURE	A procedure identifier

Table 7-35: Tool -- Attributes

7.5.4.3. Subflow

The Activity is refined as a subflow. The subflow may be executed synchronously or asynchronously. The subflow identifiers used are inherited from the surrounding Package declaration.

In the case of ***asynchronous execution*** the execution of the Activity is continued after a process instance of the referenced Process Definition is initiated (in this case execution proceeds to any post activity split logic after subflow initiation. No return parameters are supported from such called processes. Synchronization with the initiated subflow, if required, has to be done by other means such as events, not described in this document. This style of subflow is characterized as chained (or forked) subflow operation.

In the case of ***synchronous execution*** the execution of the Activity is suspended after a process instance of the

referenced Process Definition is initiated. After execution termination of this process instance the Activity is resumed. Return parameters may be used between the called and calling processes on completion of the subflow. This style of subflow is characterized as hierarchic subflow operation.

```
<xsd:element name="SubFlow">
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element ref="xpdl:ActualParameters" minOccurs="0"/>
      </xsd:sequence>
      <xsd:attribute name="Id" type="xsd:string" use="required"/>
      <xsd:attribute name="Execution">
         <xsd:simpleType>
            <xsd:restriction base="xsd:NMTOKEN">
               <xsd:enumeration value="ASYNCHR"/>
               <xsd:enumeration value="SYNCHR"/>
            </xsd:restriction>
         </xsd:simpleType>
      </xsd:attribute>
   </xsd:complexType>
</xsd:element>
```

	Description	
Actual Parameters	A list of parameters to be passed to the subflow. See section 7.1.2.3.	
Execution	ASYNCHR	Executed asynchronously.
	SYNCHR	Executed synchronously.
Id	Used to identify the workflow process that is invoked.	

Table 7-37: Subflow -- Attributes

7.5.5. Performer Relationship

The relationship of the Activity to a (potential) performer is given by the Participant Assignment attribute. It provides a link to the entity Workflow Participant. Default: Any Participant.

The Workflow Participant identifiers used in the Performer attribute have either to be declared in the surrounding Workflow Process definition or are inherited from the surrounding Package declaration.

The question whether the expression evaluation results in an empty set of performers or a non unique performer is to be handled by the workflow management system at run time or, where defined, by the external resource repository or organizational model. The runtime resolution of both cases is outside the scope of this specification

- In the first case (empty set) the engine may e.g. retry at a later time, or it may signal this to the supervisor of the process. The approach used is local to the WFMS and does not form part of this specification.
- The second case (non-unique) may arise where the performer definition is by function/skill type (defined as "Role") and/or is an organization unit, which is itself a container for a set of participants. In these situations the approach adopted for participant assignment is local to the WFMS and does not form part of this specification. Common scenarios are:
    - Where an activity includes multiple work items that may be implemented in parallel, separate work items may be presented to a number of performers.
    - In other situations the activity may be assigned according to a local load-balancing algorithm or presented to multiple potential performers in their work lists and assigned to the first accepting participant. (It is the responsibility of the workflow engine to provide the appropriate behavior.)
    - The assignment of an activity to an organizational unit (e.g. a department) may result in the activity being offered to all members of the organizational unit and assigned to the first accepting participant or allow the manager of the unit to redirect the activity to a designated departmental member.

In all cases the participant assignments defined within the meta-model and expressed in XPDL only relate Activities to defined Participants (including the use of expressions and defined Functions) and do not differentiate between cases

where the defined Participant is atomic (e.g. a person) or not (e.g. a team). The local behavior of the workflow engine and the resource repository or organizational model in handling these situations is not defined.

7.5.6. Deadline

```
<xsd:element name="Deadline">
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element name="DeadlineCondition" minOccurs="1" maxOccurs="1"/>
         <xsd:element name="ExceptionName" minOccurs="1" maxOccurs="1"/>
      </xsd:sequence>
      <xsd:attribute name="Execution">
         <xsd:simpleType>
            <xsd:restriction base="xsd:NMTOKEN">
               <xsd:enumeration value="ASYNCHR"/>
               <xsd:enumeration value="SYNCHR"/>
            </xsd:restriction>
         </xsd:simpleType>
      </xsd:attribute>
   </xsd:complexType>
</xsd:element>
```

	Description	
Execution	Define the system behaviour on raising the arrival of the deadline	
	ASYNCHR	The deadline is to be raised asynchronously. This is an implicit AND SPLIT operation, where the activity continues and another thread is started following the named exception transition. Another deadline may occur on the same activity, because it continues running.
	SYNCHR	The activity is completed abnormally and flow continues on the named exception transition.
DeadlineCondition	An expression indicating the time of the deadline. This expression is implementation dependent and may include at least: Times relative to the beginning of the activity. (2 days) Fixed times (January 1) or (January 1, 2002) Times computed using workflow relevant data (varName days)	
ExceptionName	The name of the exception to be raised on arrival of the deadline.	

Table 7-39:Deadline

Deadlines are used to raise an exception upon the expiration of a specific period of time.

Upon the arrival of a deadline, an exception condition is raised and the appropriate exception transitions are followed. If the deadline is synchronous, then the activity is terminated before flow continues on the exception path. If the deadline is asynchronous, then an implicit AND SPLIT is performed, and a new thread of processing is started on the appropriate exception transition. Asynchronous exceptions can cause side effects, and should be used carefully. Some of these side effects are discussed later in this section.

A sample deadline is below. In the sample, an asynchronous “notifyException” will be raised after 3 days. The activity will continue normally at this point. If the activity is still executing after 5 days it will be terminated and a “timeoutException” will be raised.

Sample Deadline

```
<Deadline Execution="ASYNCHR">
```

```
            <DeadlineCondition>3 days</DeadlineCondition>
            <ExceptionName>notifyException</ExceptionName>
        </Deadline>
            <Deadline Execution="SYNCHR">
            <DeadlineCondition>5 days</DeadlineCondition>
            <ExceptionName>timeoutException</ExceptionName>
        </Deadline>
```

The syntax of the deadline conditions is implementation dependent. The condition may be relative or absolute and may use workflow relevant data.

If a synchronous deadline occurs on a block activity or a subflow, stopping the activity includes stopping all the threads in the block, or the subflow and all its threads and nested subflows as well. From a modeling perspective, this can be treated as "immediate termination." If an engine chooses to deviate from this, such as allowing an in-process manual activity to complete, it should document this behavior.

An asynchronous exception can be a powerful tool, allowing intermediate notification and graceful process termination by altering workflow relevant data. But an asynchronous exception can also create race conditions and possible side effects. For instance, the running activity could complete while the asynchronous exception is being processed. In addition, because an implicit split is performed, flow control can be complicated if the asynchronous exception processing joins back up with the deadlined processing thread. Care must be taken by the workflow designer to properly handle race conditions and avoid unwanted side effects.

7.5.7. Simulation Information

```
<xsd:element name="SimulationInformation">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element ref="xpdl:Cost"/>
            <xsd:element ref="xpdl:TimeEstimation"/>
        </xsd:sequence>
        <xsd:attribute name="Instantiation">
            <xsd:simpleType>
                <xsd:restriction base="xsd:NMTOKEN">
                    <xsd:enumeration value="ONCE"/>
                    <xsd:enumeration value="MULTIPLE"/>
                </xsd:restriction>
            </xsd:simpleType>
        </xsd:attribute>
    </xsd:complexType>
</xsd:element>

<xsd:element name="TimeEstimation">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element ref="xpdl:WaitingTime" minOccurs="0"/>
            <xsd:element ref="xpdl:WorkingTime" minOccurs="0"/>
            <xsd:element ref="xpdl:Duration" minOccurs="0"/>
        </xsd:sequence>
    </xsd:complexType>
</xsd:element>

<xsd:element name="WaitingTime" type="xsd:string"/>

<xsd:element name="WorkingTime" type="xsd:string"/>

<xsd:element name="Duration" type="xsd:string"/>

<xsd:element name="Cost" type="xsd:string"/>
```

The Instantiation Attribute defines how many times an Activity can be activated for higher throughput (e.g. how many individuals can capture a role). This can be once or many times (multiple).

	Description
Cost	Average cost.
Duration	Expected duration time to perform a task in units of DurationUnit.
Instantiation	Defines the capability of an activity to be activated: once or many times (multiple) ONCE The Activity can only be instantiated once. Default. MULTIPLE The Activity can be instantiated multiple times.
Time Estimation	Expected duration (summary of working time, waiting time, and duration) in units of DurationUnit.
Waiting Time	Average waiting time in units of DurationUnit.
Working Time	Average working time in units of DurationUnit.

Table 7-40: Simulation Information -- Attributes

7.5.8. Transition Restrictions

```
<xsd:element name="TransitionRestriction">
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element ref=“xpdl:Join" minOccurs="0"/>
         <xsd:element ref=“xpdl:Split" minOccurs="0"/>
      </xsd:sequence>
   </xsd:complexType>
</xsd:element>

<xsd:element name="TransitionRestrictions">
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element ref=“xpdl:TransitionRestriction"
            minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
   </xsd:complexType>
</xsd:element>
```

	Description
Join	Specifies that the incoming Transitions of the Activity are JOIN -ed
Split	Specifies that the outgoing Transitions of the Activity are SPLIT -ed

Table 7-42: Transition Restrictions -- Attributes

7.5.8.1. Join

A join describes the semantics of an activity with multiple incoming Transitions.

```
<xsd:element name="Join">
   <xsd:complexType>
      <xsd:attribute name="Type">
         <xsd:simpleType>
            <xsd:restriction base="xsd:NMTOKEN">
               <xsd:enumeration value="AND"/>
               <xsd:enumeration value="XOR"/>
            </xsd:restriction>
         </xsd:simpleType>
      </xsd:attribute>
   </xsd:complexType>
```

```
</xsd:element>
```

	Description	
Type	AND	Join of (all) concurrent threads within the process instance with incoming transitions to the activity: Synchronization is required. The number of threads to be synchronized might be dependent on the result of the conditions of previous AND split(s).
	XOR	Join for alternative threads: No synchronisation is required.

Table 7-45: Join -- Attributes

The AND join can be seen as a " rendezvous precondition" of the Activity; the activity is not initiated until the transition conditions on all incoming routes evaluate true.

The XOR join initiates the Activity when the transition conditions of any (one) of the incoming transitions evaluates true.

7.5.8.2. Split

A split describes the semantics where multiple outgoing Transitions for an Activity exist.

```
<xsd:element name="Split">
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element ref=“xpdl:TransitionRefs" minOccurs="0"/>
      </xsd:sequence>
      <xsd:attribute name="Type">
         <xsd:simpleType>
            <xsd:restriction base="xsd:NMTOKEN">
               <xsd:enumeration value="AND"/>
               <xsd:enumeration value="XOR"/>
            </xsd:restriction>
         </xsd:simpleType>
      </xsd:attribute>
   </xsd:complexType>
</xsd:element>

<xsd:element name="TransitionRef">
   <xsd:complexType>
      <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
   </xsd:complexType>
</xsd:element>

<xsd:element name="TransitionRefs">
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element ref=“xpdl:TransitionRef"
            minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
   </xsd:complexType>
</xsd:element>
```

	Description
Transition Refs	A list of outgoing transitions from the Activity. Each transition is identified by its Id.

	Description	
Type	AND	Defines a number of possible concurrent threads represented by the outgoing Transitions of this Activity. If the Transitions have conditions the actual number of executed parallel threads is dependent on the conditions associated with each transition, which are evaluated concurrently.
	XOR	List of Identifiers of outgoing Transitions of this Activity, representing. Alternatively executed transitions. The decision as to which single transition route is selected is dependent on the conditions of each individual transition as they are evaluated in the sequence specified in the list. If an unconditional Transition is evaluated or transition with condition OTHERWISE this ends the list evaluation.

Table 7-47: Split -- Attributes

An AND split with transitions having conditions may be referred to as "conditional AND", "multiple-choice OR", or "nonexclusive OR", respectively. The number of actual concurrent threads is determined at execution time when evaluating the conditions. Following such an AND split the process instance (or thread of the process instance) is forked into a number of separate execution threads which result from the transitions condition evaluation. (Note that no list of identifiers is required since all outgoing transitions from the activity are evaluated and no sequence is necessary.)

If within the AND_SPLIT there is a transition having condition OTHERWISE, then a two-step evaluation is performed. In the first step evaluation is made of all the Transitions except that within the OTHERWISE condition. If none of them (including those having no condition) evaluate to TRUE, then in the second step the same procedure is performed for the Transition with OTHERWISE (only one transition with an OTHERWISE clause is permitted in the list of outgoing transitions from an activity).

An OTHERWISE alternative can be used to guarantee that there is no undefined status from the Process execution (i.e. at least one outgoing transition from an activity will always occur).

7.5.9. Conformance Classes

There are Conformance Classes restricting the Activity-Transition Net.

The following Conformance Classes are defined in the package:

- NON-BLOCKED
  - There is no restriction for this class.
- LOOP-BLOCKED
  - The Activities and Transitions of a Process Definition form an acyclic graph (or set of disjoint acyclic graphs).
- FULL-BLOCKED
  - For each join (or respectively split) there is exactly one corresponding split (or respectively join) of the same kind. In an AND split no conditions are permitted; in a XOR split an unconditional or OTHERWISE Transition is required if there is a Transition with a condition (i.e. an undefined result of transition evaluation is not permitted).

7.6. Transition Information

The Transition Information describes possible transitions between activities and the conditions that enable or disable them (the transitions) during workflow execution. Further control and structure restrictions may be expressed in the Activity definition.

A process definition is seen as a network of edges between the Activity nodes (i.e. as a workflow process diagram). All

edges are directed and given by a pair of Activities:

(From node, to node).

The edges of the Activity net may be labelled by ***Transition conditions***. A Transition condition for a specific edge enables that transition if the condition evaluat es to TRUE. If no routing condition is specified the Transition behaves as if a condition with value TRUE is present.

If there are multiple incoming or outgoing ("regular", see below) Transitions of an Activity, then further options to express control flow restrictions and condition evaluation semantics are provided in the Activity entity definition (AND/XOR variants of SPLIT/JOIN).

A loop may be represented via a transition that returns to an Activity that was on a path that led to the transition. Typically, at least one of the activities in the loop will have multiple outgoing transitions, one or more of which will contain an exit conditon from the loop.

For the identifiers and names defined in the Transition information the scope is the surrounding Workflow Process Definition.

It is possible to define or synchronize multiple (concurrent or alternative) control threads (split, join) and sequences of Transitions between Activities (cascading Transitions/conditions).

```
<xsd:element name="Transition">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element ref=“xpdl:Condition" minOccurs="0"/>
            <xsd:element ref=“xpdl:Description" minOccurs="0"/>
            <xsd:element ref=“xpdl:ExtendedAttributes" minOccurs="0"/>
        </xsd:sequence>
        <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
        <xsd:attribute name="From" type="xsd:NMTOKEN" use="required"/>
        <xsd:attribute name="To" type="xsd:NMTOKEN" use="required"/>
        <xsd:attribute name="Name" type="xsd:string"/>
    </xsd:complexType>
</xsd:element>

<xsd:element name="Transitions">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element ref=“xpdl:Transition" minOccurs="0"
maxOccurs="unbounded"/>
        </xsd:sequence>
    </xsd:complexType>
</xsd:element>
```

	Description
Condition	A Transition condition expression based on workflow relevant data. (E.g. 'Contract' = 'SMALL' OR 'Contract' <$20,000). Default: TRUE
Description	Short textual description of the Transition.
Extended Attributes	Optional extensions to meet individual implementation needs
From	Determines the FROM source of a Transition. (Activity Identifier)
Id	Used to identify the Transition.
Name	Text used to identify the Transition.
To	Determines the TO target of a Transition (Activity Identifier)

Table 7-49: Transition Information -- Attributes

7.6.1. Condition

```
<xsd:element name="Condition">
   <xsd:complexType mixed="true">
      <xsd:choice minOccurs="0" maxOccurs="unbounded">
         <xsd:element ref="xpdl:Xpression"/>
      </xsd:choice>
      <xsd:attribute name="Type">
         <xsd:simpleType>
            <xsd:restriction base="xsd:NMTOKEN">
               <xsd:enumeration value="CONDITION"/>
               <xsd:enumeration value="OTHERWISE"/>
               <xsd:enumeration value="EXCEPTION"/>
               <xsd:enumeration value="DEFAULTEXCEPTION"/>
            </xsd:restriction>
         </xsd:simpleType>
      </xsd:attribute>
   </xsd:complexType>
</xsd:element>

<xsd:element name="Xpression">
   <xsd:complexType mixed="true">
      <xsd:choice minOccurs="0" maxOccurs="unbounded">
         <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
      </xsd:choice>
   </xsd:complexType>
</xsd:element>
```

	Description	
Type	Define the type of transition condition, valid values are	
	CONDITION	Indicates that the transition is to be executed if its condition is satisfied,
	OTHERWISE	Indicates that the transition is the default transition that is executed if no conditions are met.
	EXCEPTION	Indicates that the transition is to be executed if there is an exception and its condition is satisfied.
	DEFAULTEXCEPTION	Indicates that the transition is the default transition that is executed if there is an exception and no exception conditions are met.
Xpression	A condition expression represented via XML markup.	

Table 7-51: Condition -- Attributes

7.6.1.1. Exception Conditions

The Exception and DEFAULTEXCEPTION types allow you to specify branches that are taken only when an Exception is raised. The EXCEPTION type is equivalent to the CONDITION type and the DEFAULTEXCEPTION matches the OTHERWISE type. The Condition may contain either the name of an Exception or a more complex expression. Except for the deadlines, exceptions are raised in an engine-specific manner. Like regular transitions, exception transitions are traversed only after the "From" activity has completed. An exception usually indicates abnormal completion.

The following example illustrates a set of transitions from an activity that includes exceptions: branches 1 and 2 are processed under normal conditions; branches 3 and 4 are processed if there is an exception.

```
<Transitions>
   <Transition Id="branch1" From="CheckBalance" To="ProcessRequest">
      <Condition Type="CONDITION">Balance > 1000</Condition>
   </Transition>
   <Transition Id="branch2" From="CheckBalance" To="InsufFunds">
      <Condition Type="OTHERWISE"/>
```

```
    </Transition>
    <Transition Id="branch3" From="CheckBalance" To="MessageDisplay">
        <Condition Type="EXCEPTION">ATMDownException</Condition>
    </Transition>
    <Transition Id="branch4" From="CheckBalance" To="SendAlarm">
        <Condition Type="DEFAULTEXCEPTION"/>
    </Transition>
</Transitions>
```

7.7. Workflow Participants

The Workflow Participant is one of the following types: resource set, resource, organizational unit, role, human, or system. A role and a resource are used in the sense of abstract actors. . This definition is an abstraction level between the real performer and the activity, which has to be performed. During run time these abstract definitions are evaluated and assigned to concrete human(s) and/or program(s).

The scope of the identifier of a workflow participant entity declaration in a minimal resource repository or organizational model is the surrounding entity (Workflow Process Definition or Process Model Definition) in which it is defined.

An external resource repository or organizational model may contain substantial additional information that complements the basic participant types presented in here.

```
    <xsd:element name="Participant">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element ref=“xpdl:ParticipantType"/>
                <xsd:element ref=“xpdl:Description" minOccurs="0"/>
                <xsd:element ref=“xpdl:ExternalReference" minOccurs="0"/>

                <xsd:element ref=“xpdl:ExtendedAttributes" minOccurs="0"/>
            </xsd:sequence>
            <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
            <xsd:attribute name="Name" type="xsd:string"/>
        </xsd:complexType>
    </xsd:element>

    <xsd:element name="Participants">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element ref=“xpdl:Participant" minOccurs="0"
maxOccurs="unbounded"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
```

	Description
Description	Short textual description of a workflow participant.
ExternalReference	A reference to an external specification of a participant. See section 7.1.3
Extended Attributes	Optional extensions to meet individual implementation needs
Id	Used to identify the workflow participant definition.
Name	Text used to identify a performer
Participant Type	Definition of the type of workflow participant entity.

Table 7-53: Workflow Participant -- Attributes

7.7.1. Participant Entity Types

The Participant entity type attribute characterises the participant to be an individual, an organisational unit or an abstract resource such as a machine.

```
<xsd:element name="ParticipantType">
   <xsd:complexType>
      <xsd:attribute name="Type" use="required">
         <xsd:simpleType>
            <xsd:restriction base="xsd:NMTOKEN">
               <xsd:enumeration value="RESOURCE_SET"/>
               <xsd:enumeration value="RESOURCE"/>
               <xsd:enumeration value="ROLE"/>
               <xsd:enumeration value="ORGANIZATIONAL_UNIT"/>
               <xsd:enumeration value="HUMAN"/>
               <xsd:enumeration value="SYSTEM"/>
            </xsd:restriction>
         </xsd:simpleType>
      </xsd:attribute>
   </xsd:complexType>
</xsd:element>
```

	Description	
Type	RESOURCE_SET	A set of resources.
	RESOURCE	A specific resource agent.
	ROLE	This type allows performer addressing by a role or skill set. A role in this context is a function a human has within an organization. As a function isn't necessarily unique, a coordinator may be defined (for administrative purposes or in case of exception handling) and a list of humans the role is related to.
	ORGANIZATIONAL_UNIT	A department or any other unit within an organizational model.
	HUMAN	A human interacting with the system via an application presenting a user interface to the participant.
	SYSTEM	An automatic agent.

Table 7-55: Participant Entity Type -- Attributes

7.8. Workflow Relevant Data

Workflow relevant data represent the variables of a workflow process or Package Definition. They are typically used to maintain decision data (used in conditions) or reference data values (parameters), which are passed between activities or subflow. This may be differentiated from workflow application data, which is data managed or accessed wholly by the invoked applications and which is not accessible to the workflow management system. The workflow relevant data list defines all data objects, which are required by the workflow process. The attribute `DataType` explicitly specifies all information needed for a workflow management system to define an appropriate data object for storing data, which is to be handled by an active instance of the workflow process.

Workflow relevant data can be defined in a workflow process and in a Package. The scopes differ in that the former may only be accessed by entities defined inside that process, while the latter may be used also e.g. to define the parameters of a process entity.

Where parameters are passed to a called subflow outside the current model definition (e.g. to support remote process

invocation) it is the responsibility of the process designer(s) to ensure that data type compatibility exists across the parameter set.

```
<xsd:element name="DataField">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element ref=“xpdl:DataType"/>
            <xsd:element ref=“xpdl:InitialValue" minOccurs="0"/>
            <xsd:element ref=“xpdl:Length" minOccurs="0"/>
            <xsd:element ref=“xpdl:Description" minOccurs="0"/>
            <xsd:element ref=“xpdl:ExtendedAttributes" minOccurs="0"/>
        </xsd:sequence>
        <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
        <xsd:attribute name="Name" type="xsd:string"/>
        <xsd:attribute name="IsArray" default="FALSE">
            <xsd:simpleType>
                <xsd:restriction base="xsd:NMTOKEN">
                    <xsd:enumeration value="TRUE"/>
                    <xsd:enumeration value="FALSE"/>
                </xsd:restriction>
            </xsd:simpleType>
        </xsd:attribute>
    </xsd:complexType>
</xsd:element>

<xsd:element name="DataFields">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element ref=“xpdl:DataField" minOccurs="0"
maxOccurs="unbounded"/>
        </xsd:sequence>
    </xsd:complexType>
</xsd:element>

<xsd:element name="DataType">
    <xsd:complexType>
        <xsd:group ref=”xpdl:DataTypes”/>
    </xsd:complexType>
</xsd:element>


<xsd:element name="InitialValue" type="xsd:string"/>

<xsd:element name="Length" type="xsd:string"/>
```

	Description
Data Type	Data type of the process variable. See Section 7.9
Description	Short textual description of the data defined.
Extended Attributes	Optional extensions to meet individual implementation needs
Id	Used to identify the workflow relevant data.
Initial Value	Pre-assignment of data for run time.
Is Array	Indicates if it is an array
Length	The length of the data
Name	Text used to identify the workflow relevant data

Table 7-57: Workflow Relevant Data -- Attributes

7.9. Data Types

Data types consist of a set of standard types that may be used as part of the data specification of workflow relevant data, formal parameters, and Workflow processes. You can also declare a new data type within a TypeDeclaration and use it wherever the standard datatypes are used. A data type may be selected from the following set of types.

```
<xsd:group name="DataTypes">
   <xsd:choice>
      <xsd:element ref="xpdl:BasicType"/>
      <xsd:element ref="xpdl:DeclaredType"/>
      <xsd:element ref="xpdl:SchemaType"/>
      <xsd:element ref="xpdl:ExternalReference"/>
      <xsd:element ref="xpdl:RecordType"/>
      <xsd:element ref="xpdl:UnionType"/>
      <xsd:element ref="xpdl:EnumerationType"/>
      <xsd:element ref="xpdl:ArrayType"/>
      <xsd:element ref="xpdl:ListType"/>
      </xsd:choice>
</xsd:group>
```

	Description
Array Type	A fixed size set of data all of the same datatype (deprecated).
Basic Type	A simple type: STRING, INTEGER, FLOAT, DATETIME, REFERENCE, BOOLEAN, or PERFORMER.
Declared Type	A reference to a data type declared in a TypeDeclaration element.
Enumeration Type	A set of legal values of a variable or parameter (deprecated).
ExternalReference	A reference to a type defined in an external document. See Section 7.1.3
List Type	An unbounded set of data all of the same data type (deprecated).
Record Type	A set of members that may be of different types (deprecated).
SchemaType	A data type defined using an XML schema.
Union Type	A set of members only one of which will be used for an instance of the data (deprecated).

Table 7-59: Standard Data Types

7.9.1. Basic Data Types

```
<xsd:element name="BasicType">
   <xsd:complexType>
      <xsd:attribute name="Type" use="required">
         <xsd:simpleType>
            <xsd:restriction base="xsd:NMTOKEN">
               <xsd:enumeration value="STRING"/>
               <xsd:enumeration value="FLOAT"/>
               <xsd:enumeration value="INTEGER"/>
               <xsd:enumeration value="REFERENCE"/>
               <xsd:enumeration value="DATETIME"/>
               <xsd:enumeration value="BOOLEAN"/>
               <xsd:enumeration value="PERFORMER"/>
            </xsd:restriction>
         </xsd:simpleType>
      </xsd:attribute>
   </xsd:complexType>
</xsd:element>
```

	Description	
Type	STRING	A finite-length sequence of characters
	FLOAT	A floating point or double precision number. The maximum size of the number is not specified in XPDL
	INTEGER	A number represented by an optional sign followed by a finite-length sequence of decimal digits. The maximum size of the integer is not specified in XPDL
	REFERENCE	A reference to an external data type – now deprecated. The ExternalReference is the recommended way to refer to an external data type.
	DATETIME	A specific instant of time. The date format is not specified within XPDL.
	BOOLEAN	A data instance of a Boolean type is one having one of the values TRUE or FALSE. The internal representation of these values is not defined in the XPDL
	PERFORMER	A data instance of a performer type is one having a value of a declared workflow participant.

Table 7-61: Basic Data Types -- Attributes

7.9.2. Complex Data Types

XPDL permits the definition of complex data types such as arrays, records, unions, enumerations, and lists. Complex data types are defined using the SchemaType. The RecordType, UnionType, EnumerationType, ArrayType, and ListType, which were used in the past to define complex data, are now deprecated. They have been left in the xpdl schema for compatibility with previous versions.

7.9.2.1. Schema Type

The SchemaType allows users to define a data type using XML schema syntax. It may also be used to define an XML string that should conform to the schema.

```
<xsd:element name="SchemaType">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xsd:sequence>
    </xsd:complexType>
</xsd:element>
```

The following, for example, could describe a C++ or Java class, a C struct, or an XML string:

```
<SchemaType>
    <schema xmlns="http://www.w3.org/2000/10/XMLSchema">
        <element name="PO">
            <complexType>
                <sequence>
                    <element name="CustomerName" type="string"/>
                    <element name="Address" type="string"/>
                    <element name="OrderNumber" type="string"/>
                </sequence>
            </complexType>
        </element>
    </schema>
</SchemaType>
```

7.9.2.2. Record Type

```
<xsd:element name="RecordType">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element ref=“xpdl:Member" maxOccurs="unbounded"/>
        </xsd:sequence>
    </xsd:complexType>
</xsd:element>

<xsd:element name="Member">
    <xsd:complexType>
        <xsd:group ref="xpdl:DataTypes"/>
    </xsd:complexType>
</xsd:element>
```

	Description
Member	A field in the record.
DataTypes	Data type of a member. See Table 7-59: Standard Data Types.

Table 7-63: Record Type – Attributes

7.9.2.3. Union Type

```
<xsd:element name="UnionType">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element ref=“xpdl:Member" maxOccurs="unbounded"/>
        </xsd:sequence>
    </xsd:complexType>
</xsd:element>

<xsd:element name="Member">
    <xsd:complexType>
        <xsd:group ref="xpdl:DataTypes"/>
    </xsd:complexType>
</xsd:element>
```

	Description
Member	A field in the union.
DataTypes	Data type of a member. See Table 7-59: Standard Data Types.

Table 7-65: Union Type

7.9.2.4. Enumeration Type

```
<xsd:element name="EnumerationType">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element ref=“xpdl:EnumerationValue" maxOccurs="unbounded"/>
        </xsd:sequence>
    </xsd:complexType>
</xsd:element>

<xsd:element name="EnumerationValue">
    <xsd:complexType>
        <xsd:attribute name="Name" type="xsd:NMTOKEN" use="required"/>
```

```
    </xsd:complexType>
</xsd:element>
```

	Description
Enumeration Value	An element that represents one of the values in an enumeration.
Name	The name of the value.

Table 7-67: Enumeration Type -- Attributes

7.9.2.5. Array Type

```
<xsd:element name="ArrayType">
    <xsd:complexType>
        <xsd:group ref=”xpdl:DataTypes”>
        </xsd:group>
        <xsd:attribute name="LowerIndex" type="xsd:NMTOKEN" use="required"/>
        <xsd:attribute name="UpperIndex" type="xsd:NMTOKEN" use="required"/>
    </xsd:complexType>
</xsd:element>
```

	Description
DataTypes	The data type of array entries. See Table 7-59: Standard Data Types.
Lower Index	The lower bound of an ArrayType.
Upper Index	The upper bound of an ArrayType.

Table 7-69: Array Type -- Attributes

7.9.2.6. List Type

```
<xsd:element name="ListType">
    <xsd:complexType>
        <xsd:group ref=”xpdl:DataTypes”>
        </xsd:group>
    </xsd:complexType>
</xsd:element>
```

	Description
DataTypes	The data type of list entries. See Table 7-59: Standard Data Types.

Table 7-71: List Type -- Attributes

7.9.3. Declared Data Types

It is possible to reuse a complex data definition wherever you can use a standard xpdl type. Define the data type under a TypeDeclaration and then refer to it using the DeclaredType data type.

7.9.3.1. Type Declaration

```
<xsd:element name="TypeDeclaration">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:group ref=”xpdl:DataTypes”>
            <xsd:element ref=“xpdl:Description" minOccurs="0"/>
            <xsd:element ref=“xpdl:ExtendedAttributes" minOccurs="0"/>
        </xsd:sequence>
        <xsd:attribute name="Id" type="xsd:ID" use="required"/>
```

```
            <xsd:attribute name="Name" type="xsd:string"/>
        </xsd:complexType>
    </xsd:element>

    <xsd:element name="TypeDeclarations">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element ref="xpdl:TypeDeclaration"
                    minOccurs="0" maxOccurs="unbounded"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
```

	Description
DataTypes	The data type. See Table 7-59: Standard Data Types.
Description	An informal description of the data type.
ExtendedAttributes	Optional extensions to meet individual implementation needs.
Id	An identifier for the TypeDeclaration.
Name	The name of the TypeDeclaration.

Table 7-73: Type Declaration

To reuse a SchemaType to define a purchase order:

```
    <TypeDeclarations>
        <TypeDeclaration Id="POType" Name="PurchaseOrder">
            <SchemaType>
                <schema xmlns="http://www.w3.org/2000/10/XMLSchema">
                    <element name="PO">
                        <complexType>
                            <sequence>
                              <element name="CustomerName" type="string"/>
                              <element name="Address" type="string"/>
                              <element name="OrderNumber" type="string"/>
                            </sequence>
                        </complexType>
                    </element>
                </schema>
            </SchemaType>
        </TypeDeclaration>
    </TypeDeclarations>
```

7.9.3.2. Declared Type

```
    <xsd:element name="DeclaredType">
        <xsd:complexType>
            <xsd:attribute name="Id" type="xsd:IDREF" use="required"/>
        </xsd:complexType>
    </xsd:element>
```

	Description
Id	A reference to a data type declared in a TypeDeclaration.

Table 7-7438: Declared Data Type – Attributes

Using the DeclaredType type you could, for example, define a number of POType variables.

```
    <DataFields>
        <DataField Id="newPO">
```

```
            <DataType>
               <DeclaredType Id="POType"/>
            </DataType>
         </DataField>
         <DataField Id="checkedPO">
            <DataType>
               <DeclaredType Id="POType"/>
            </DataType>
         </DataField>
      </DataFields>
```

8. Sample Workflow

This sample workflow was created to illustrate some of the features of XPDL and does not represent any real processes or necessarily the best way to accomplish the process. The XPDL in the example was generated by a Visio add-on that was created by CapeVisions and edited to incorporate unsupported features and improve readability.

The workflow represents an order entry system in which an order enters the system as a formatted string. The Package is composed of a main process and two subprocesses. The following discussion includes a brief overview of these processes along with a discussion of some of the data types, extended attributes, and external references used in the sample. Finally the XPDL is displayed.

8.1. The Processes

8.1.1. The EOrder Main Process

The main process takes a formatted string as an input and returns a string that indicates whether the order was confirmed or rejected. It contains the following steps:

- The string is first converted to a complex data object. If an exception is caught (indicating that the string is incorrectly formatted), an alarm is raised and the order is rejected.
- The data is checked for accuracy.
- The process determines whether payment is via a purchase order or a credit card.
- Credit card orders are sent to a subprocess that authorizes the credit purchase.
- Purchase orders are validated by an application that checks the vendor's record and authorizes the purchase amount.
- The order is entered into the database and an order number is issued.
- The next three activities happen in parallel:

  An acceptance message is composed to return to the end user.

  A subprocess is invoked asynchronously to fill the order.

  An order confirmation email is sent to the end user. This activity is a special activity that is managed by the workflow system. It uses ExtendedAttributes to specify the information the system needs for the email.
- If an order is rejected, either because it is inaccurate or cannot obtain authorization, a rejection message is composed, to return to the customer.

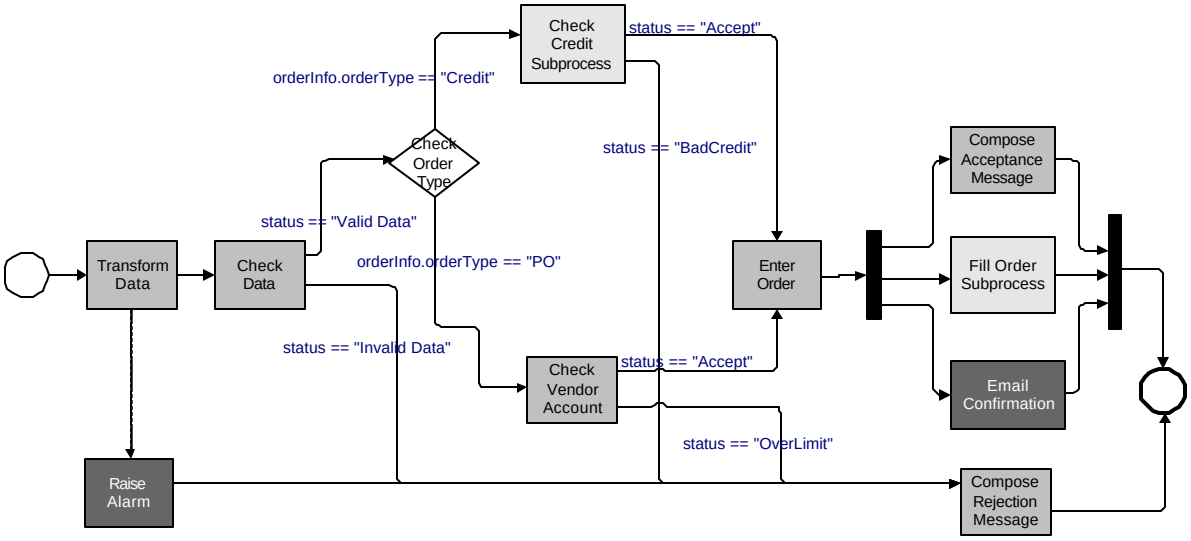

Figure 8-1: EOrder Main Process

8.1.2. The CreditCheck Subprocess

The CreditCheck subprocess sets up a CreditInfo object from the input parameters and then sends the information to a credit card web service for authorization. The web service returns a status string that is converted to an OrderStatus string and returned to the calling process.

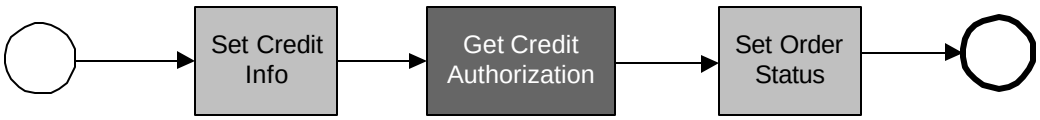

Figure 8-2: CreditCheck Subprocess

8.1.3. The FillOrder Subprocess

This subprocess handles the shipping and billing of the order. This process includes a participant called a "Shipper"

- The first activity displays the order information to a Shipper who ships the items in the order and records the status of the line items. The application returns the status of the order -- whether it is complete or backordered. This activity includes some deadlines. If the activity is not completed within 3 days, a notifyException is thrown an alarm is raised. If the activity is still not completed within 5 days, a timeoutException is thrown and the order is canceled.
- The process then determines if it is a PO or credit order.
- PO orders are sent to the billing system and then an electronic invoice is created and stored on a server.
- Credit card orders are sent to the credit card web service for charging and then an electronic receipt is created and stored on a server.
- The last step sends the invoice or receipt to the customer as an attachment to an email message. It uses ExtendedAttributes to specify the information needed for the email.

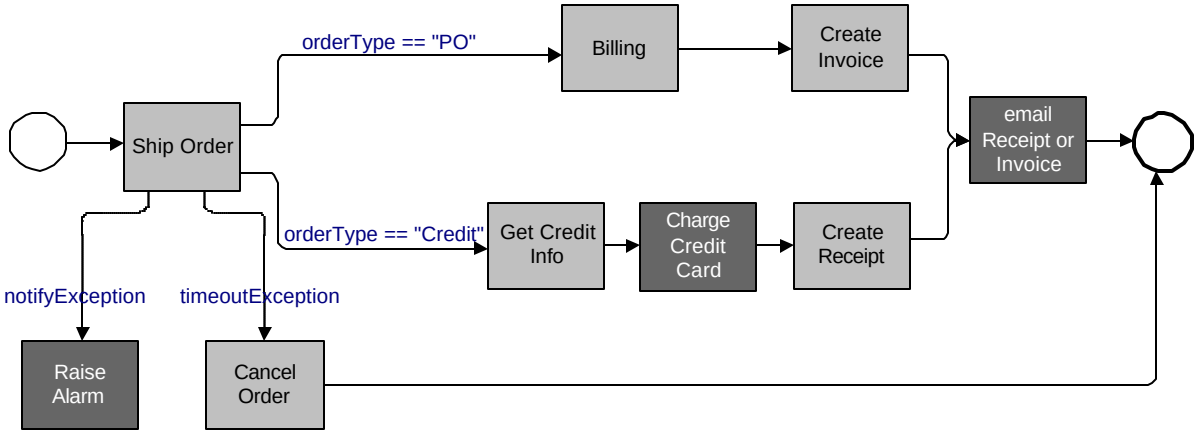

Figure 8-3: FillOrder Subprocess

8.2. Type Declarations

A number of data types are defined for the workflow.

- An Order is defined in a separate schema document and declared using an ExternalReference.

```
<TypeDeclaration Id="Order" Name="Order">
   <ExternalReference
location="http://wfmc.org/standards/docs/xpdl_sample/orderschema.xsd"/>
</TypeDeclaration>
```

The schema is:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified" attributeFormDefault="unqualified">
   <xsd:element name="Order">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="Items">
               <xsd:complexType>
                  <xsd:sequence>
                     <xsd:element name="Item" maxOccurs="unbounded">
                       <xsd:complexType>
                           <xsd:attribute name="itemNumber" type="xsd:integer"
use="required"/>
                           <xsd:attribute name="itemQty" type="xsd:integer"
use="required"/>
                       </xsd:complexType>
                     </xsd:element>
                  </xsd:sequence>
               </xsd:complexType>
            </xsd:element>
         </xsd:sequence>
         <xsd:attribute name="accountNumber" type="xsd:integer" use="required"/>
         <xsd:attribute name="totalAmount" type="xsd:float" use="required"/>
         <xsd:attribute name="emailAddress" type="xsd:string" use="required"/>
         <xsd:attribute name="orderType" use="required">
            <xsd:simpleType>
               <xsd:restriction base="xsd:NMTOKEN">
                  <xsd:enumeration value="PO"/>
                  <xsd:enumeration value="Credit"/>
               </xsd:restriction>
```

```
            </xsd:simpleType>
        </xsd:attribute>
        <xsd:attribute name="cardType" use="required">
            <xsd:simpleType>
                <xsd:restriction base="xsd:NMTOKEN">
                    <xsd:enumeration value="MC-VISA"/>
                    <xsd:enumeration value="Discover"/>
                    <xsd:enumeration value="AMEX"/>
                </xsd:restriction>
            </xsd:simpleType>
        </xsd:attribute>
    </xsd:complexType>
</xsd:element>
</xsd:schema>
```

- An OrderStatus is directly defined using a SchemaType. It uses an XML schema to enumerate the valid strings that can represent the status.

```
<TypeDeclaration Id="OrderStatus" Name="OrderStatus">
    <SchemaType>
        <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified" attributeFormDefault="unqualified">
            <xsd:element name="Status">
                <xsd:simpleType>
                    <xsd:restriction base="xsd:NMTOKEN">
                        <xsd:enumeration value="ValidData"/>
                            <xsd:enumeration value="InvalidData"/>
                            <xsd:enumeration value="Accept"/>
                            <xsd:enumeration value="BadCredit"/>
                            <xsd:enumeration value="OverLimit"/>
                            <xsd:enumeration value="BadDataFormat"/>
                        </xsd:restriction>
                    </xsd:simpleType>
                </xsd:element>
            </xsd:schema>
        </SchemaType>
</TypeDeclaration>
```

- A CardType type uses an ExternalReference to the cardType attribute within the Order schema

```
    <TypeDeclaration Id="CardType" Name="CardType">
        <ExternalReference
location="http://wfmc.org/standards/docs/xpdl_sample/orderschema.xsd"
xref="cardType" namespace="orderschema/Order"/>
    </TypeDeclaration>
```

- A CreditInfo type uses an ExternalReference to a data type defined within a WSDL document.

```
    <TypeDeclaration Id="CreditInfo" Name="CreditInfo">
        <ExternalReference
location="http://wfmc.org/standards/docs/xpdl_sample/creditService.wsdl"
xref="CreditInfo"/>
    </TypeDeclaration>
```

Within the WSDL document, the type is defined as follows:

```
    <types>
        <schema xmlns="http://www.w3.org/2001/XMLSchema">
            <element name="CreditInfo">
                <complexType>
                    <sequence>
                        <element name="MerchantNumber"/>
```

```
                    <element name="AccountNumber"/>
                    <element name="Amount"/>
                    <element name="CardType"/>
                </sequence>
            </complexType>
        </element>
    </schema>
</types>
```

8.3. ExtendedAttributes

The workflow vendor defines several ExtendedAttributes to extend XPDL. The namespace, xmlns:xyz="http://www.xyzeorder.com/workflow", is designated for the ExtendedAttribute XML.

- There is an ExtendedAttribute to mark an activity as a SystemActivity, or one implemented by the workflow system. Three activities in the sample are so marked: email activities, alarm activities, and web service activities.

```
<ExtendedAttribute Name="SystemActivity" Value="WebService"/>
<ExtendedAttribute Name="SystemActivity" Value="Email"/>
<ExtendedAttribute Name="SystemActivity" Value="Alarm"/>
```

- There is an ExtendedAttribute to indicate the position of an activity.

```
<ExtendedAttribute Name="Coordinates">
    <xyz:Coordinates xpos="381" ypos="316"/>
</ExtendedAttribute>
```

- There is an ExtendedAttribute to provide information for the email activity. It should be assumed that some of this information is supplied in the process modeling tool. Note that the workflow vendor supports the capability of referencing workflow relevant data within strings by prefacing a variable name with “%%”.

```
<ExtendedAttribute Name="Email">
    <xyz:Email to="%%emailAddress" subject="%%orderStatus">
        <xyz:Attachments>
            <xyz:Attachment>%%docURI</xyz:Attachment>
        </xyz:Attachments>
        <xyz:MessageText>Order number %%orderNumber is %%orderStatus. Thank-you
for ordering from PQR Products, Inc.</xyz:MessageText>
    </xyz:Email>
</ExtendedAttribute>
```

8.4. External References

The sample workflow makes an external reference to a WSDL document, creditService.wsdl, to define the applications used for processing a credit card purchase. (Section 8.2illustrates the use of the ExternalReference element to import data types from external documents.)

```
<Application Id="getCreditAuthorization">
   <ExternalReference
location="http://wfmc.org/standards/docs/xpdl_sample/creditService.wsdl"
xref="GetCreditAuthorization"/>
</Application>
```

Within the WSDL document the applications are defined as operations:

```
<message name="creditInput">
   <part name="CreditInfo" element="tns:CreditInfo"/>
```

```
</message>
<message name="creditOutput">
   <part name="status" type="string"/>
</message>
<portType name="CreditPortType">
   <operation name="GetCreditAuthorization">
      <input message="tns:creditInput"/>
      <output message="tns:creditOutput"/>
   </operation>
   <operation name="ChargeCreditAccount">
         <input message="tns:creditInput"/>
      <output message="tns:creditOutput"/>
   </operation>
</portType>
```

8.5. Sample XPDL

```
<?xml version="1.0" encoding="us-ascii"?>
<Package xmlns="http://www.wfmc.org/2002/XPDL1.0"
xmlns:xpdl="http://www.wfmc.org/2002/XPDL1.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xyz="http://www.xyzeorder.com/workflow"
xsi:schemaLocation="http://www.wfmc.org/2002/XPDL1.0
http://wfmc.org/standards/docs/TC-1025_schema_10_xpdl.xsd" Id="0" Name="sample
workflow process">
   <PackageHeader>
      <XPDLVersion>0.09</XPDLVersion>
      <Vendor>XYZ, Inc</Vendor>
      <Created>6/18/2002 5:27:17 PM</Created>
   </PackageHeader>
   <ConformanceClass GraphConformance="NON_BLOCKED"/>
   <Script Type="text/javascript"/>
   <TypeDeclarations>
      <TypeDeclaration Id="Order" Name="Order">
         <ExternalReference
location="http://wfmc.org/standards/docs/xpdl_sample/orderschema.xsd"/>
      </TypeDeclaration>
      <TypeDeclaration Id="CreditInfo" Name="CreditInfo">
         <ExternalReference
location="http://wfmc.org/standards/docs/xpdl_sample/creditService.wsdl"
xref="CreditInfo"/>
      </TypeDeclaration>
      <TypeDeclaration Id="CardType" Name="CardType">
         <ExternalReference
location="http://wfmc.org/standards/docs/xpdl_sample/orderschema.xsd"
xref="cardType" namespace="orderschema/Order"/>
      </TypeDeclaration>
      <TypeDeclaration Id="OrderStatus" Name="OrderStatus">
         <SchemaType>
            <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified" attributeFormDefault="unqualified">
               <xsd:element name="Status">
                  <xsd:simpleType>
                     <xsd:restriction base="xsd:NMTOKEN">
                       <xsd:enumeration value="ValidData"/>
                       <xsd:enumeration value="InvalidData"/>
                       <xsd:enumeration value="Accept"/>
                       <xsd:enumeration value="BadCredit"/>
                       <xsd:enumeration value="OverLimit"/>
                       <xsd:enumeration value="BadDataFormat"/>
                     </xsd:restriction>
                  </xsd:simpleType>
               </xsd:element>
            </xsd:schema>
```

```
            </SchemaType>
        </TypeDeclaration>
    </TypeDeclarations>
    <Participants>
        <Participant Id="DBConnection">
            <ParticipantType Type="SYSTEM"/>
            <Description>Reference to Database Resource</Description>
        </Participant>
    </Participants>
    <Applications/>
    <DataFields/>
    <WorkflowProcesses>
        <WorkflowProcess Id="1" Name="EOrder" AccessLevel="PUBLIC">
            <ProcessHeader/>
            <FormalParameters>
                <FormalParameter Id="orderString" Index="1" Mode="IN">
                    <DataType>
                        <BasicType Type="STRING"/>
                    </DataType>
                </FormalParameter>
                <FormalParameter Id="returnMessage" Index="2" Mode="OUT">
                    <DataType>
                        <BasicType Type="STRING"/>
                    </DataType>
                </FormalParameter>
            </FormalParameters>
            <DataFields>
                <DataField Id="orderNumber" IsArray="FALSE">
                    <DataType>
                        <BasicType Type="INTEGER"/>
                    </DataType>
                    <InitialValue>0</InitialValue>
                    <Length>0</Length>
                </DataField>
                <DataField Id="status" IsArray="FALSE">
                    <DataType>
                        <DeclaredType Id="OrderStatus"/>
                    </DataType>
                </DataField>
                <DataField Id="orderInfo" IsArray="FALSE">
                    <DataType>
                        <DeclaredType Id="Order"/>
                    </DataType>
                </DataField>
            </DataFields>
            <Participants/>
            <Applications>
                <Application Id="transformData">
                    <FormalParameters>
                        <FormalParameter Id="orderStringIn" Index="1" Mode="IN">
                            <DataType>
                              <BasicType Type="STRING"/>
                            </DataType>
                        </FormalParameter>
                        <FormalParameter Id="orderInfo" Index="2" Mode="OUT">
                            <DataType>
                              <DeclaredType Id="Order"/>
                            </DataType>
                        </FormalParameter>
                    </FormalParameters>
                </Application>
                <Application Id="checkData">
                    <FormalParameters>
                        <FormalParameter Id="orderInfo" Index="1" Mode="IN">
                            <DataType>
                              <DeclaredType Id="Order"/>
                            </DataType>
```

```
                </FormalParameter>
                <FormalParameter Id="statusOut" Index="2" Mode="OUT">
                    <DataType>
                      <DeclaredType Id="OrderStatus"/>
                    </DataType>
                </FormalParameter>
            </FormalParameters>
        </Application>
        <Application Id="checkVendor">
            <FormalParameters>
                <FormalParameter Id="accountNumberIn" Index="1" Mode="IN">
                    <DataType>
                      <BasicType Type="INTEGER"/>
                    </DataType>
                </FormalParameter>
                <FormalParameter Id="amountIn" Index="2" Mode="IN">
                    <DataType>
                      <BasicType Type="FLOAT"/>
                    </DataType>
                </FormalParameter>
                <FormalParameter Id="statusOut" Index="3" Mode="OUT">
                    <DataType>
                      <DeclaredType Id="OrderStatus"/>
                    </DataType>
                </FormalParameter>
            </FormalParameters>
        </Application>
        <Application Id="enterOrder">
            <FormalParameters>
                <FormalParameter Id="orderInfoIn" Index="1" Mode="IN">
                    <DataType>
                      <DeclaredType Id="Order"/>
                    </DataType>
                </FormalParameter>
                <FormalParameter Id="orderNumber" Index="2" Mode="OUT">
                    <DataType>
                      <BasicType Type="INTEGER"/>
                    </DataType>
                </FormalParameter>
            </FormalParameters>
        </Application>
        <Application Id="composeMessage">
            <FormalParameters>
                <FormalParameter Id="statusIn" Index="1" Mode="IN">
                    <DataType>
                      <DeclaredType Id="OrderStatus"/>
                    </DataType>
                </FormalParameter>
                <FormalParameter Id="orderNumber" Index="2" Mode="IN">
                    <DataType>
                      <BasicType Type="INTEGER"/>
                    </DataType>
                </FormalParameter>
            </FormalParameters>
        </Application>
    </Applications>
    <Activities>
        <Activity Id="1" Name="Check Data">
            <Implementation>
                <Tool Id="checkData" Type="APPLICATION">
                    <ActualParameters>
                      <ActualParameter>orderInfo</ActualParameter>
                      <ActualParameter>status</ActualParameter>
                    </ActualParameters>
                </Tool>
            </Implementation>
            <TransitionRestrictions>
```

```
                        <TransitionRestriction>
                            <Split Type="XOR">
                              <TransitionRefs>
                                  <TransitionRef Id="22"/>
                                  <TransitionRef Id="23"/>
                              </TransitionRefs>
                            </Split>
                        </TransitionRestriction>
                    </TransitionRestrictions>
                    <ExtendedAttributes>
                        <ExtendedAttribute Name="Coordinates">
                            <xyz:Coordinates xpos="183" ypos="389"/>
                        </ExtendedAttribute>
                    </ExtendedAttributes>
                </Activity>
                <Activity Id="5">
                    <Route/>
                    <ExtendedAttributes>
                        <ExtendedAttribute Name="Coordinates">
                            <xyz:Coordinates xpos="35" ypos="389"/>
                        </ExtendedAttribute>
                    </ExtendedAttributes>
                </Activity>
                <Activity Id="6">
                    <Route/>
                    <TransitionRestrictions>
                        <TransitionRestriction>
                            <Join Type="XOR"/>
                        </TransitionRestriction>
                    </TransitionRestrictions>
                    <ExtendedAttributes>
                        <ExtendedAttribute Name="Coordinates">
                            <xyz:Coordinates xpos="755" ypos="315"/>
                        </ExtendedAttribute>
                    </ExtendedAttributes>
                </Activity>
                <Activity Id="8" Name="Email Confirmation">
                    <Implementation>
                        <No/>
                    </Implementation>
                    <ExtendedAttributes>
                        <ExtendedAttribute Name="Coordinates">
                            <xyz:Coordinates xpos="657" ypos="312"/>
                        </ExtendedAttribute>
                        <ExtendedAttribute Name="SystemActivity" Value="Email"/>
                        <ExtendedAttribute Name="Email">
                            <xyz:Email to="%%orderInfo.emailAddress" subject="Order
%%orderNumber">
                              <xyz:MessageText>Order number %%orderNumber is being
processed. Thank-you for ordering from PQR Products, Inc</xyz:MessageText>
                            </xyz:Email>
                        </ExtendedAttribute>
                    </ExtendedAttributes>
                </Activity>
                <Activity Id="9">
                    <Route/>
                    <TransitionRestrictions>
                        <TransitionRestriction>
                            <Split Type="AND">
                              <TransitionRefs>
                                  <TransitionRef Id="1"/>
                                  <TransitionRef Id="38"/>
                                  <TransitionRef Id="2"/>
                              </TransitionRefs>
                            </Split>
                        </TransitionRestriction>
                    </TransitionRestrictions>
```

```
                    <ExtendedAttributes>
                        <ExtendedAttribute Name="Coordinates">
                            <xyz:Coordinates xpos="572" ypos="389"/>
                        </ExtendedAttribute>
                    </ExtendedAttributes>
                </Activity>
                <Activity Id="10" Name="Check Credit Subprocess">
                    <Implementation>
                        <SubFlow Id="3" Execution="SYNCHR">
                            <ActualParameters/>
                        </SubFlow>
                    </Implementation>
                    <TransitionRestrictions>
                        <TransitionRestriction>
                            <Split Type="XOR">
                              <TransitionRefs>
                                   <TransitionRef Id="26"/>
                                   <TransitionRef Id="31"/>
                              </TransitionRefs>
                            </Split>
                        </TransitionRestriction>
                    </TransitionRestrictions>
                    <ExtendedAttributes>
                        <ExtendedAttribute Name="Coordinates">
                            <xyz:Coordinates xpos="381" ypos="535"/>
                        </ExtendedAttribute>
                    </ExtendedAttributes>
                </Activity>
                <Activity Id="11" Name="Fill Order
Subprocess">
                    <Implementation>
                        <SubFlow Id="2" Execution="ASYNCHR">
                            <ActualParameters>
                              <ActualParameter>orderNumber</ActualParameter>
                              <ActualParameter>orderInfo.orderType</ActualParameter>
                              <ActualParameter>orderInfo.emailAddress</ActualParameter>
                            </ActualParameters>
                        </SubFlow>
                    </Implementation>
                    <ExtendedAttributes>
                        <ExtendedAttribute Name="Coordinates">
                            <xyz:Coordinates xpos="653" ypos="389"/>
                        </ExtendedAttribute>
                    </ExtendedAttributes>
                </Activity>
                <Activity Id="12" Name="Check Order Type">
                    <Route/>
                    <TransitionRestrictions>
                        <TransitionRestriction>
                            <Split Type="XOR">
                              <TransitionRefs>
                                   <TransitionRef Id="24"/>
                                   <TransitionRef Id="25"/>
                              </TransitionRefs>
                            </Split>
                        </TransitionRestriction>
                    </TransitionRestrictions>
                    <ExtendedAttributes>
                        <ExtendedAttribute Name="Coordinates">
                            <xyz:Coordinates xpos="293" ypos="460"/>
                        </ExtendedAttribute>
                    </ExtendedAttributes>
                </Activity>
                <Activity Id="17" Name="Transform Data">
                    <Implementation>
                        <Tool Id="transformData" Type="APPLICATION">
                            <ActualParameters>
```

```
                <ActualParameter>orderString</ActualParameter>
                <ActualParameter>orderInfo</ActualParameter>
            </ActualParameters>
        </Tool>
    </Implementation>
    <TransitionRestrictions>
        <TransitionRestriction>
            <Split Type="XOR">
              <TransitionRefs>
                  <TransitionRef Id="40"/>
                  <TransitionRef Id="21"/>
              </TransitionRefs>
            </Split>
        </TransitionRestriction>
    </TransitionRestrictions>
    <ExtendedAttributes>
        <ExtendedAttribute Name="Coordinates">
            <xyz:Coordinates xpos="102" ypos="389"/>
        </ExtendedAttribute>
    </ExtendedAttributes>
</Activity>
<Activity Id="32" Name="Enter Order">
    <Implementation>
        <Tool Id="enterOrder" Type="APPLICATION">
            <ActualParameters>
              <ActualParameter>orderInfo</ActualParameter>
              <ActualParameter>orderNumber</ActualParameter>
            </ActualParameters>
        </Tool>
    </Implementation>
    <Performer>DBConnection</Performer>
    <TransitionRestrictions>
        <TransitionRestriction>
            <Join Type="XOR"/>
        </TransitionRestriction>
    </TransitionRestrictions>
    <ExtendedAttributes>
        <ExtendedAttribute Name="Coordinates">
            <xyz:Coordinates xpos="510" ypos="389"/>
        </ExtendedAttribute>
    </ExtendedAttributes>
</Activity>
<Activity Id="33">
    <Route/>
    <TransitionRestrictions>
        <TransitionRestriction>
            <Join Type="AND"/>
        </TransitionRestriction>
    </TransitionRestrictions>
    <ExtendedAttributes>
        <ExtendedAttribute Name="Coordinates">
            <xyz:Coordinates xpos="725" ypos="391"/>
        </ExtendedAttribute>
    </ExtendedAttributes>
</Activity>
<Activity Id="39" Name="Compose RejectionMessage">
    <Implementation>
        <Tool Id="composeMessage" Type="APPLICATION">
            <ActualParameters>
              <ActualParameter>orderNumber</ActualParameter>
              <ActualParameter>-1</ActualParameter>
            </ActualParameters>
        </Tool>
    </Implementation>
    <TransitionRestrictions>
        <TransitionRestriction>
            <Join Type="XOR"/>
```

```
                    </TransitionRestriction>
                </TransitionRestrictions>
                <ExtendedAttributes>
                    <ExtendedAttribute Name="Coordinates">
                        <xyz:Coordinates xpos="655" ypos="245"/>
                    </ExtendedAttribute>
                </ExtendedAttributes>
            </Activity>
            <Activity Id="41" Name="Check Vendor Account">
                <Implementation>
                    <Tool Id="checkVendor" Type="APPLICATION">
                        <ActualParameters>

<ActualParameter>orderInfo.AccountNumber</ActualParameter>
                          <ActualParameter>orderInfo.ToltalAmount</ActualParameter>
                          <ActualParameter>status</ActualParameter>
                        </ActualParameters>
                    </Tool>
                </Implementation>
                <Performer>DBConnection</Performer>
                <TransitionRestrictions>
                    <TransitionRestriction>
                        <Split Type="XOR">
                          <TransitionRefs>
                              <TransitionRef Id="27"/>
                              <TransitionRef Id="30"/>
                          </TransitionRefs>
                        </Split>
                    </TransitionRestriction>
                </TransitionRestrictions>
                <ExtendedAttributes>
                    <ExtendedAttribute Name="Coordinates">
                        <xyz:Coordinates xpos="381" ypos="316"/>
                    </ExtendedAttribute>
                </ExtendedAttributes>
            </Activity>
            <Activity Id="56" Name="Compose Acceptance Message">
                <Implementation>
                    <Tool Id="composeMessage" Type="APPLICATION">
                        <ActualParameters>
                          <ActualParameter>status</ActualParameter>
                          <ActualParameter>orderNumber</ActualParameter>
                        </ActualParameters>
                    </Tool>
                </Implementation>
                <ExtendedAttributes>
                    <ExtendedAttribute Name="Coordinates">
                        <xyz:Coordinates xpos="653" ypos="462"/>
                    </ExtendedAttribute>
                </ExtendedAttributes>
            </Activity>
            <Activity Id="58" Name="Raise Alarm">
                <Implementation>
                    <No/>
                </Implementation>
                <ExtendedAttributes>
                    <ExtendedAttribute Name="Coordinates">
                        <xyz:Coordinates xpos="100" ypos="250"/>
                    </ExtendedAttribute>
                    <ExtendedAttribute Name="SystemActivity" Value="Alarm"/>
                </ExtendedAttributes>
            </Activity>
        </Activities>
        <Transitions>
            <Transition Id="1" From="9" To="8"/>
            <Transition Id="2" From="9" To="11"/>
            <Transition Id="16" From="11" To="33"/>
```

```
                <Transition Id="17" From="8" To="33">
                   <Condition Type="OTHERWISE"/>
                </Transition>
                <Transition Id="18" From="33" To="6"/>
                <Transition Id="20" From="5" To="17"/>
                <Transition Id="21" From="17" To="1"/>
                <Transition Id="22" From="1" To="12">
                   <Condition>status == "Valid Data"</Condition>
                </Transition>
                <Transition Id="23" From="1" To="39">
                   <Condition>status == "Invalid Data"</Condition>
                </Transition>
                <Transition Id="24" From="12" To="10">
                   <Condition>orderType == "Credit"</Condition>
                </Transition>
                <Transition Id="25" From="12" To="41">
                   <Condition>orderType == "PO"</Condition>
                </Transition>
                <Transition Id="26" From="10" To="32">
                   <Condition>status == "Accept"</Condition>
                </Transition>
                <Transition Id="27" From="41" To="32">
                   <Condition>status == "Accept"</Condition>
                </Transition>
                <Transition Id="28" From="32" To="9"/>
                <Transition Id="29" From="39" To="6"/>
                <Transition Id="30" From="41" To="39">
                   <Condition>status == "OverLimit"</Condition>
                </Transition>
                <Transition Id="31" From="10" To="39">
                   <Condition>status == "BadCredit"</Condition>
                </Transition>
                <Transition Id="38" From="9" To="56"/>
                <Transition Id="39" From="56" To="33"/>
                <Transition Id="40" From="17" To="58">
                   <Condition Type="EXCEPTION"/>
                </Transition>
                <Transition Id="42" From="58" To="39"/>
             </Transitions>
          </WorkflowProcess>
          <WorkflowProcess Id="2" Name="FillOrder" AccessLevel="PRIVATE">
             <ProcessHeader/>
             <FormalParameters>
                <FormalParameter Id="orderNumber" Index="1" Mode="IN">
                   <DataType>
                      <BasicType Type="INTEGER"/>
                   </DataType>
                   <Description>Order number assigned to the order.</Description>
                </FormalParameter>
                <FormalParameter Id="orderType" Index="1" Mode="IN">
                   <DataType>
                      <ExternalReference
location="http://wfmc.org/standards/docs/xpdl_sample/orderschema.xsd"
xref="orderType" namespace="orderschema/Order"/>
                   </DataType>
                </FormalParameter>
                <FormalParameter Id="emailAddress" Index="1" Mode="IN">
                   <DataType>
                      <BasicType Type="STRING"/>
                   </DataType>
                </FormalParameter>
             </FormalParameters>
             <DataFields>
                <DataField Id="docURI" IsArray="FALSE">
                   <DataType>
                      <BasicType Type="STRING"/>
                   </DataType>
```

```
                    <Description>URI of receipt or invoice.</Description>
                </DataField>
                <DataField Id="orderStatus" IsArray="FALSE">
                    <DataType>
                        <BasicType Type="STRING"/>
                    </DataType>
                </DataField>
                <DataField Id="creditInfo" IsArray="FALSE">
                    <DataType>
                        <DeclaredType Id="CreditInfo"/>
                    </DataType>
                </DataField>
                <DataField Id="creditStatus" IsArray="FALSE">
                    <DataType>
                        <BasicType Type="STRING"/>
                    </DataType>
                </DataField>
            </DataFields>
            <Participants>
                <Participant Id="Shipper">
                    <ParticipantType Type="ROLE"/>
                    <Description>Order shipper</Description>
                </Participant>
            </Participants>
            <Applications>
                <Application Id="shipOrder">
                    <Description>This application presents a screen that presents
order information for the order identified by shipOrder. The user may update the
order with any changes such as back order information. It returns a string
indicating whether the order is complete or on back order.</Description>
                    <FormalParameters>
                        <FormalParameter Id="OrderNumberParam" Index="1" Mode="IN">
                            <DataType>
                              <BasicType Type="INTEGER"/>
                            </DataType>
                        </FormalParameter>
                        <FormalParameter Id="Status" Index="2" Mode="OUT">
                            <DataType>
                              <BasicType Type="STRING"/>
                            </DataType>
                            <Description>The String that describes the status -- either
"Complete" or "Backorder"</Description>
                        </FormalParameter>
                    </FormalParameters>
                </Application>
                <Application Id="charge">
                    <Description>Charges the credit card and prepares a receipt for a
credit order</Description>
                    <ExternalReference
location="http://wfmc.org/standards/docs/xpdl_sample/creditService.wsdl"
xref="ChargeCreditAccount"/>
                </Application>
                <Application Id="billAccount">
                    <Description>Bills the vendor account</Description>
                    <FormalParameters>
                        <FormalParameter Id="orderNumberParam" Index="1" Mode="IN">
                            <DataType>
                              <BasicType Type="INTEGER"/>
                            </DataType>
                        </FormalParameter>
                    </FormalParameters>
                </Application>
                <Application Id="createInvoice">
                    <Description>Creates an invoice using the order information and
stores it on a server.</Description>
                    <FormalParameters>
                        <FormalParameter Id="orderNumber" Index="1" Mode="IN">
```

```
                        <DataType>
                          <BasicType Type="INTEGER"/>
                        </DataType>
                    </FormalParameter>
                    <FormalParameter Id="docURI" Index="2" Mode="OUT">
                        <DataType>
                          <BasicType Type="STRING"/>
                        </DataType>
                    </FormalParameter>
                </FormalParameters>
            </Application>
            <Application Id="createReceipt">
                <Description>Creates a receiptusing the order information and
stores it on a server.</Description>
                <FormalParameters>
                    <FormalParameter Id="orderNumber" Index="1" Mode="IN">
                        <DataType>
                          <BasicType Type="INTEGER"/>
                        </DataType>
                    </FormalParameter>
                    <FormalParameter Id="docURI" Index="2" Mode="OUT">
                        <DataType>
                          <BasicType Type="STRING"/>
                        </DataType>
                    </FormalParameter>
                </FormalParameters>
            </Application>
            <Application Id="cancelOrder">
                <FormalParameters>
                    <FormalParameter Id="orderNumberIn" Index="1" Mode="IN">
                        <DataType>
                          <BasicType Type="INTEGER"/>
                        </DataType>
                    </FormalParameter>
                </FormalParameters>
            </Application>
        </Applications>
        <Activities>
            <Activity Id="21">
                <Route/>
                <ExtendedAttributes>
                    <ExtendedAttribute Name="Coordinates">
                        <xyz:Coordinates xpos="62" ypos="389"/>
                    </ExtendedAttribute>
                </ExtendedAttributes>
            </Activity>
            <Activity Id="22" Name="Billing">
                <Implementation>
                    <Tool Id="billAccount" Type="APPLICATION">
                        <ActualParameters>
                          <ActualParameter>orderNumber</ActualParameter>
                        </ActualParameters>
                    </Tool>
                </Implementation>
                <Performer>DBConnection</Performer>
                <ExtendedAttributes>
                    <ExtendedAttribute Name="Coordinates">
                        <xyz:Coordinates xpos="347" ypos="435"/>
                    </ExtendedAttribute>
                </ExtendedAttributes>
            </Activity>
            <Activity Id="23" Name="Charge Credit Card">
                <Implementation>
                    <No/>
                </Implementation>
                <ExtendedAttributes>
                    <ExtendedAttribute Name="SystemActivity" Value="WebService"/>
```

```
                        <ExtendedAttribute Name="Coordinates">
                            <xyz:Coordinates xpos="386" ypos="338"/>
                        </ExtendedAttribute>
                    </ExtendedAttributes>
                </Activity>
                <Activity Id="30">
                    <Route/>
                    <ExtendedAttributes>
                        <ExtendedAttribute Name="Coordinates">
                            <xyz:Coordinates xpos="613" ypos="389"/>
                        </ExtendedAttribute>
                    </ExtendedAttributes>
                </Activity>
                <Activity Id="31" Name="email Receipt or Invoice">
                    <Implementation>
                        <No/>
                    </Implementation>
                    <TransitionRestrictions>
                        <TransitionRestriction>
                            <Join Type="XOR"/>
                        </TransitionRestriction>
                    </TransitionRestrictions>
                    <ExtendedAttributes>
                        <ExtendedAttribute Name="Coordinates">
                            <xyz:Coordinates xpos="430" ypos="385"/>
                        </ExtendedAttribute>
                        <ExtendedAttribute Name="SystemActivity" Value="Email"/>
                        <ExtendedAttribute Name="Email">
                            <xyz:Email to="%%emailAddress" subject="%%orderStatus">
                              <xyz:Attachments>
                                  <xyz:Attachment>%%docURI</xyz:Attachment>
                              </xyz:Attachments>
                              <xyz:MessageText>Order number %%orderNumber is
%%orderStatus. Thank-you for ordering from PQR Products, Inc.</xyz:MessageText>
                            </xyz:Email>
                        </ExtendedAttribute>
                    </ExtendedAttributes>
                </Activity>
                <Activity Id="36" Name="Ship Order">
                    <Description>View order and enter fulfillment info</Description>
                    <Implementation>
                        <Tool Id="shipOrder" Type="APPLICATION">
                            <ActualParameters>
                              <ActualParameter>orderNumber</ActualParameter>
                              <ActualParameter>orderStatus</ActualParameter>
                            </ActualParameters>
                        </Tool>
                    </Implementation>
                    <Performer>DBConnection and Shipper</Performer>
                    <Deadline Execution="ASYNCHR">
                        <DeadlineCondition>3 days</DeadlineCondition>
                        <ExceptionName>notifyException</ExceptionName>
                    </Deadline>
                    <Deadline Execution="SYNCHR">
                        <DeadlineCondition>5 days</DeadlineCondition>
                        <ExceptionName>timeoutException</ExceptionName>
                    </Deadline>
                    <TransitionRestrictions>
                        <TransitionRestriction>
                            <Split Type="XOR">
                              <TransitionRefs>
                                  <TransitionRef Id="11"/>
                                  <TransitionRef Id="10"/>
                                  <TransitionRef Id="49"/>
                                  <TransitionRef Id="50"/>
                              </TransitionRefs>
                            </Split>
```

```
            </TransitionRestriction>
        </TransitionRestrictions>
        <ExtendedAttributes>
            <ExtendedAttribute Name="Coordinates">
                <xyz:Coordinates xpos="133" ypos="386"/>
            </ExtendedAttribute>
        </ExtendedAttributes>
    </Activity>
    <Activity Id="59" Name="Get Credit Info">
        <Implementation>
            <Tool Id="getCreditInfo" Type="APPLICATION">
                <ActualParameters>
                  <ActualParameter>orderNumber</ActualParameter>
                  <ActualParameter>creditInfo</ActualParameter>
                </ActualParameters>
            </Tool>
        </Implementation>
        <Performer>DBConnection</Performer>
        <ExtendedAttributes>
            <ExtendedAttribute Name="Coordinates">
                <xyz:Coordinates xpos="311" ypos="338"/>
            </ExtendedAttribute>
        </ExtendedAttributes>
    </Activity>
    <Activity Id="60" Name="Create Invoice">
        <Implementation>
            <Tool Id="createInvoice" Type="APPLICATION">
                <ActualParameters>
                  <ActualParameter>orderNumber</ActualParameter>
                  <ActualParameter>docUri</ActualParameter>
                </ActualParameters>
            </Tool>
        </Implementation>
        <Performer>DBConnection</Performer>
        <ExtendedAttributes>
            <ExtendedAttribute Name="Coordinates">
                <xyz:Coordinates xpos="460" ypos="435"/>
            </ExtendedAttribute>
        </ExtendedAttributes>
    </Activity>
    <Activity Id="61" Name="Create Receipt">
        <Implementation>
            <Tool Id="createReceipt" Type="APPLICATION">
                <ActualParameters>
                  <ActualParameter>orderNumber</ActualParameter>
                  <ActualParameter>docUri</ActualParameter>
                </ActualParameters>
            </Tool>
        </Implementation>
        <Performer>DBConnection</Performer>
        <ExtendedAttributes>
            <ExtendedAttribute Name="Coordinates">
                <xyz:Coordinates xpos="461" ypos="338"/>
            </ExtendedAttribute>
        </ExtendedAttributes>
    </Activity>
    <Activity Id="63" Name="Raise Alarm">
        <Implementation>
            <No/>
        </Implementation>
        <ExtendedAttributes>
            <ExtendedAttribute Name="Coordinates">
                <xyz:Coordinates xpos="83" ypos="271"/>
            </ExtendedAttribute>
            <ExtendedAttribute Name="SystemActivity" Value="Alarm"/>
        </ExtendedAttributes>
    </Activity>
```

```
            <Activity Id="64" Name="Cancel Order">
               <Description>View order and enter fulfillment info</Description>
               <Implementation>
                  <Tool Id="cancelOrder" Type="APPLICATION">
                     <ActualParameters>
                       <ActualParameter>orderNumber</ActualParameter>
                     </ActualParameters>
                  </Tool>
               </Implementation>
               <Performer>DBConnection</Performer>
               <ExtendedAttributes>
                  <ExtendedAttribute Name="Coordinates">
                     <xyz:Coordinates xpos="174" ypos="271"/>
                  </ExtendedAttribute>
               </ExtendedAttributes>
            </Activity>
         </Activities>
         <Transitions>
            <Transition Id="8" From="21" To="36"/>
            <Transition Id="10" From="36" To="22">
               <Condition>orderType == "PO"</Condition>
            </Transition>
            <Transition Id="11" From="36" To="59">
               <Condition>orderType == "Credit"</Condition>
            </Transition>
            <Transition Id="13" From="60" To="31"/>
            <Transition Id="14" From="61" To="31"/>
            <Transition Id="15" From="31" To="30"/>
            <Transition Id="43" From="59" To="23"/>
            <Transition Id="44" From="23" To="61"/>
            <Transition Id="45" From="22" To="60"/>
            <Transition Id="49" From="36" To="63">
               <Condition>notifyException</Condition>
            </Transition>
            <Transition Id="50" From="36" To="64">
               <Condition>timeoutException</Condition>
            </Transition>
            <Transition Id="51" From="64" To="30"/>
         </Transitions>
      </WorkflowProcess>
      <WorkflowProcess Id="3" Name="CreditCheck" AccessLevel="PRIVATE">
         <ProcessHeader/>
         <FormalParameters>
            <FormalParameter Id="accountNumber" Index="1" Mode="IN">
               <DataType>
                  <BasicType Type="INTEGER"/>
               </DataType>
            </FormalParameter>
            <FormalParameter Id="amount" Index="2" Mode="IN">
               <DataType>
                  <BasicType Type="FLOAT"/>
               </DataType>
            </FormalParameter>
            <FormalParameter Id="cardType" Index="4" Mode="IN">
               <DataType>
                  <DeclaredType Id="CardType"/>
               </DataType>
            </FormalParameter>
            <FormalParameter Id="status" Index="3" Mode="OUT">
               <DataType>
                  <DeclaredType Id="OrderStatus"/>
               </DataType>
            </FormalParameter>
         </FormalParameters>
         <DataFields>
            <DataField Id="creditStatus" IsArray="FALSE">
               <DataType>
```

```
                    <BasicType Type="STRING"/>
                </DataType>
                <Length>0</Length>
            </DataField>
        </DataFields>
        <Participants/>
        <Applications>
            <Application Id="setCreditInfo">
                <Description>Creates and initializes a CreditInfo
object.</Description>
                <FormalParameters>
                    <FormalParameter Id="accountNumber" Index="1" Mode="IN">
                        <DataType>
                          <BasicType Type="INTEGER"/>
                        </DataType>
                    </FormalParameter>
                    <FormalParameter Id="amount" Index="2" Mode="IN">
                        <DataType>
                          <BasicType Type="FLOAT"/>
                        </DataType>
                    </FormalParameter>
                    <FormalParameter Id="cardType" Index="3" Mode="IN">
                        <DataType>
                          <DeclaredType Id="CardType"/>
                        </DataType>
                    </FormalParameter>
                    <FormalParameter Id="creditInfo" Index="4" Mode="OUT">
                        <DataType>
                          <DeclaredType Id="CreditInfo"/>
                        </DataType>
                    </FormalParameter>
                </FormalParameters>
            </Application>
            <Application Id="getCreditAuthorization">
                <Description>Gets credit authorization from a charge card web
service.</Description>
                <ExternalReference
location="http://wfmc.org/standards/docs/xpdl_sample/creditService.wsdl"
xref="GetCreditAuthorization"/>
            </Application>
            <Application Id="setOrderStatus">
                <Description>Converts status returned by credit check to
OrderStatus.</Description>
                <FormalParameters>
                    <FormalParameter Id="creditStatus" Index="1" Mode="IN">
                        <DataType>
                          <BasicType Type="STRING"/>
                        </DataType>
                    </FormalParameter>
                    <FormalParameter Id="orderStatus" Index="2" Mode="OUT">
                        <DataType>
                          <DeclaredType Id="OrderStatus"/>
                        </DataType>
                    </FormalParameter>
                </FormalParameters>
            </Application>
        </Applications>
        <Activities>
            <Activity Id="48">
                <Route/>
                <ExtendedAttributes>
                    <ExtendedAttribute Name="Coordinates">
                        <xyz:Coordinates xpos="61" ypos="395"/>
                    </ExtendedAttribute>
                </ExtendedAttributes>
            </Activity>
            <Activity Id="49" Name="Set Credit Info">
```

```
                    <Implementation>
                        <Tool Id="setCreditInfo" Type="APPLICATION">
                            <ActualParameters>
                              <ActualParameter>accountNumber</ActualParameter>
                              <ActualParameter>amount</ActualParameter>
                              <ActualParameter>cardType</ActualParameter>
                              <ActualParameter>creditInfo</ActualParameter>
                            </ActualParameters>
                        </Tool>
                    </Implementation>
                    <Performer>DBConnection</Performer>
                    <ExtendedAttributes>
                        <ExtendedAttribute Name="Coordinates">
                            <xyz:Coordinates xpos="151" ypos="394"/>
                        </ExtendedAttribute>
                    </ExtendedAttributes>
                </Activity>
                <Activity Id="50" Name="Get Credit Authorization">
                    <Implementation>
                        <Tool Id="getCreditAuthorization" Type="APPLICATION">
                            <ActualParameters>
                              <ActualParameter>creditInfo</ActualParameter>
                              <ActualParameter>creditStatus</ActualParameter>
                            </ActualParameters>
                        </Tool>
                    </Implementation>
                    <ExtendedAttributes>
                        <ExtendedAttribute Name="SystemActivity" Value="WebService"/>
                        <ExtendedAttribute Name="Coordinates">
                            <xyz:Coordinates xpos="253" ypos="394"/>
                        </ExtendedAttribute>
                    </ExtendedAttributes>
                </Activity>
                <Activity Id="52">
                    <Route/>
                    <ExtendedAttributes>
                        <ExtendedAttribute Name="Coordinates">
                            <xyz:Coordinates xpos="444" ypos="397"/>
                        </ExtendedAttribute>
                    </ExtendedAttributes>
                </Activity>
                <Activity Id="62" Name="Set Order Status">
                    <Implementation>
                        <Tool Id="setOrderStatus" Type="APPLICATION">
                            <ActualParameters>
                              <ActualParameter>creditStatus</ActualParameter>
                              <ActualParameter>status</ActualParameter>
                            </ActualParameters>
                        </Tool>
                    </Implementation>
                    <ExtendedAttributes>
                        <ExtendedAttribute Name="Coordinates">
                            <xyz:Coordinates xpos="361" ypos="394"/>
                        </ExtendedAttribute>
                    </ExtendedAttributes>
                </Activity>
            </Activities>
            <Transitions>
                <Transition Id="35" From="50" To="62"/>
                <Transition Id="46" From="48" To="49"/>
                <Transition Id="47" From="49" To="50"/>
                <Transition Id="48" From="62" To="52"/>
            </Transitions>
        </WorkflowProcess>
    </WorkflowProcesses>
</Package>
```


9. XPDL Schema

This section presents the full Schema for XPDL.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema targetNamespace="http://www.wfmc.org/2002/XPDL1.0"
xmlns:xpdl="http://www.wfmc.org/2002/XPDL1.0"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified">
   <xsd:element name="Activities">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:Activity" minOccurs="0" maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="Activity">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:Description" minOccurs="0"/>
            <xsd:element ref="xpdl:Limit" minOccurs="0"/>
            <xsd:choice>
               <xsd:element ref="xpdl:Route"/>
               <xsd:element ref="xpdl:Implementation"/>
               <xsd:element ref="xpdl:BlockActivity"/>
            </xsd:choice>
            <xsd:element ref="xpdl:Performer" minOccurs="0"/>
            <xsd:element ref="xpdl:StartMode" minOccurs="0"/>
            <xsd:element ref="xpdl:FinishMode" minOccurs="0"/>
            <xsd:element ref="xpdl:Priority" minOccurs="0"/>
            <xsd:element ref="xpdl:Deadline" minOccurs="0" maxOccurs="unbounded"/>
            <xsd:element ref="xpdl:SimulationInformation" minOccurs="0"/>
            <xsd:element ref="xpdl:Icon" minOccurs="0"/>
            <xsd:element ref="xpdl:Documentation" minOccurs="0"/>
            <xsd:element ref="xpdl:TransitionRestrictions" minOccurs="0"/>
            <xsd:element ref="xpdl:ExtendedAttributes" minOccurs="0"/>
         </xsd:sequence>
         <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
         <xsd:attribute name="Name" type="xsd:string"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="ActivitySet">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:Activities" minOccurs="0"/>
            <xsd:element ref="xpdl:Transitions" minOccurs="0"/>
         </xsd:sequence>
         <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="ActivitySets">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:ActivitySet" minOccurs="0"
maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="ActualParameter" type="xsd:string"/>
   <xsd:element name="ActualParameters">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:ActualParameter" minOccurs="0"
maxOccurs="unbounded"/>
```

```
            </xsd:sequence>
         </xsd:complexType>
      </xsd:element>
      <xsd:element name="Application">
         <xsd:complexType>
            <xsd:sequence>
               <xsd:element ref="xpdl:Description" minOccurs="0"/>
               <xsd:choice>
                  <xsd:element ref="xpdl:FormalParameters"/>
                  <xsd:element ref="xpdl:ExternalReference" minOccurs="0"/>
               </xsd:choice>
               <xsd:element ref="xpdl:ExtendedAttributes" minOccurs="0"/>
            </xsd:sequence>
            <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
            <xsd:attribute name="Name" type="xsd:string"/>
         </xsd:complexType>
      </xsd:element>
      <xsd:element name="Applications">
         <xsd:complexType>
            <xsd:sequence>
               <xsd:element ref="xpdl:Application" minOccurs="0"
maxOccurs="unbounded"/>
            </xsd:sequence>
         </xsd:complexType>
      </xsd:element>
      <xsd:element name="ArrayType">
         <xsd:complexType>
            <xsd:group ref="xpdl:DataTypes"/>
            <xsd:attribute name="LowerIndex" type="xsd:NMTOKEN" use="required"/>
            <xsd:attribute name="UpperIndex" type="xsd:NMTOKEN" use="required"/>
         </xsd:complexType>
      </xsd:element>
      <xsd:element name="Author" type="xsd:string"/>
      <xsd:element name="Automatic">
         <xsd:complexType/>
      </xsd:element>
      <xsd:element name="BasicType">
         <xsd:complexType>
            <xsd:attribute name="Type" use="required">
               <xsd:simpleType>
                  <xsd:restriction base="xsd:NMTOKEN">
                     <xsd:enumeration value="STRING"/>
                     <xsd:enumeration value="FLOAT"/>
                     <xsd:enumeration value="INTEGER"/>
                     <xsd:enumeration value="REFERENCE"/>
                     <xsd:enumeration value="DATETIME"/>
                     <xsd:enumeration value="BOOLEAN"/>
                     <xsd:enumeration value="PERFORMER"/>
                  </xsd:restriction>
               </xsd:simpleType>
            </xsd:attribute>
         </xsd:complexType>
      </xsd:element>
      <xsd:element name="BlockActivity">
         <xsd:complexType>
            <xsd:attribute name="BlockId" type="xsd:NMTOKEN" use="required"/>
         </xsd:complexType>
      </xsd:element>
      <xsd:element name="Codepage" type="xsd:string"/>
      <xsd:element name="Condition">
         <xsd:complexType mixed="true">
            <xsd:choice minOccurs="0" maxOccurs="unbounded">
               <xsd:element ref="xpdl:Xpression"/>
            </xsd:choice>
            <xsd:attribute name="Type">
               <xsd:simpleType>
                  <xsd:restriction base="xsd:NMTOKEN">
```

```
                  <xsd:enumeration value="CONDITION"/>
                  <xsd:enumeration value="OTHERWISE"/>
                  <xsd:enumeration value="EXCEPTION"/>
                  <xsd:enumeration value="DEFAULTEXCEPTION"/>
               </xsd:restriction>
            </xsd:simpleType>
         </xsd:attribute>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="ConformanceClass">
      <xsd:complexType>
         <xsd:attribute name="GraphConformance">
            <xsd:simpleType>
               <xsd:restriction base="xsd:NMTOKEN">
                  <xsd:enumeration value="FULL_BLOCKED"/>
                  <xsd:enumeration value="LOOP_BLOCKED"/>
                  <xsd:enumeration value="NON_BLOCKED"/>
               </xsd:restriction>
            </xsd:simpleType>
         </xsd:attribute>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="Cost" type="xsd:string"/>
   <xsd:element name="CostUnit" type="xsd:string"/>
   <xsd:element name="Countrykey" type="xsd:string"/>
   <xsd:element name="Created" type="xsd:string"/>
   <xsd:element name="DataField">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:DataType"/>
            <xsd:element ref="xpdl:InitialValue" minOccurs="0"/>
            <xsd:element ref="xpdl:Length" minOccurs="0"/>
            <xsd:element ref="xpdl:Description" minOccurs="0"/>
            <xsd:element ref="xpdl:ExtendedAttributes" minOccurs="0"/>
         </xsd:sequence>
         <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
         <xsd:attribute name="Name" type="xsd:string"/>
         <xsd:attribute name="IsArray" default="FALSE">
            <xsd:simpleType>
               <xsd:restriction base="xsd:NMTOKEN">
                  <xsd:enumeration value="TRUE"/>
                  <xsd:enumeration value="FALSE"/>
               </xsd:restriction>
            </xsd:simpleType>
         </xsd:attribute>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="DataFields">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:DataField" minOccurs="0"
maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="DataType">
      <xsd:complexType>
         <xsd:group ref="xpdl:DataTypes"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:group name="DataTypes">
      <xsd:choice>
         <xsd:element ref="xpdl:BasicType"/>
         <xsd:element ref="xpdl:DeclaredType"/>
         <xsd:element ref="xpdl:SchemaType"/>
         <xsd:element ref="xpdl:ExternalReference"/>
         <xsd:element ref="xpdl:RecordType"/>
```

```
            <xsd:element ref="xpdl:UnionType"/>
            <xsd:element ref="xpdl:EnumerationType"/>
            <xsd:element ref="xpdl:ArrayType"/>
            <xsd:element ref="xpdl:ListType"/>
         </xsd:choice>
      </xsd:group>
      <xsd:element name="Deadline">
         <xsd:complexType>
            <xsd:sequence>
               <xsd:element name="DeadlineCondition" minOccurs="1" maxOccurs="1"/>
               <xsd:element name="ExceptionName" minOccurs="1" maxOccurs="1"/>
            </xsd:sequence>
            <xsd:attribute name="Execution">
               <xsd:simpleType>
                  <xsd:restriction base="xsd:NMTOKEN">
                     <xsd:enumeration value="ASYNCHR"/>
                     <xsd:enumeration value="SYNCHR"/>
                  </xsd:restriction>
               </xsd:simpleType>
            </xsd:attribute>
         </xsd:complexType>
      </xsd:element>
      <xsd:element name="DeclaredType">
         <xsd:complexType>
            <xsd:attribute name="Id" type="xsd:IDREF" use="required"/>
         </xsd:complexType>
      </xsd:element>
      <xsd:element name="Description" type="xsd:string"/>
      <xsd:element name="Documentation" type="xsd:string"/>
      <xsd:element name="Duration" type="xsd:string"/>
      <xsd:element name="EnumerationType">
         <xsd:complexType>
            <xsd:sequence>
               <xsd:element ref="xpdl:EnumerationValue" maxOccurs="unbounded"/>
            </xsd:sequence>
         </xsd:complexType>
      </xsd:element>
      <xsd:element name="EnumerationValue">
         <xsd:complexType>
            <xsd:attribute name="Name" type="xsd:NMTOKEN" use="required"/>
         </xsd:complexType>
      </xsd:element>
      <xsd:element name="ExtendedAttribute">
         <xsd:complexType mixed="true">
            <xsd:choice minOccurs="0" maxOccurs="unbounded">
               <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
            </xsd:choice>
            <xsd:attribute name="Name" type="xsd:NMTOKEN" use="required"/>
            <xsd:attribute name="Value" type="xsd:string"/>
         </xsd:complexType>
      </xsd:element>
      <xsd:element name="ExtendedAttributes">
         <xsd:complexType>
            <xsd:sequence>
               <xsd:element ref="xpdl:ExtendedAttribute" minOccurs="0"
   maxOccurs="unbounded"/>
            </xsd:sequence>
         </xsd:complexType>
      </xsd:element>
      <xsd:element name="ExternalPackage">
         <xsd:complexType>
            <xsd:sequence>
               <xsd:element ref="xpdl:ExtendedAttributes" minOccurs="0"/>
            </xsd:sequence>
            <xsd:attribute name="href" type="xsd:string"/>
         </xsd:complexType>
      </xsd:element>
```

```
   <xsd:element name="ExternalPackages">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:ExternalPackage" minOccurs="0"
maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="ExternalReference">
      <xsd:complexType>
         <xsd:attribute name="xref" type="xsd:NMTOKEN" use="optional"/>
         <xsd:attribute name="location" type="xsd:anyURI" use="required"/>
         <xsd:attribute name="namespace" type="xsd:anyURI" use="optional"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="FinishMode">
      <xsd:complexType>
         <xsd:choice>
            <xsd:element ref="xpdl:Automatic"/>
            <xsd:element ref="xpdl:Manual"/>
         </xsd:choice>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="FormalParameter">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:DataType"/>
            <xsd:element ref="xpdl:Description" minOccurs="0"/>
         </xsd:sequence>
         <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
         <xsd:attribute name="Index" type="xsd:NMTOKEN"/>
         <xsd:attribute name="Mode" default="IN">
            <xsd:simpleType>
               <xsd:restriction base="xsd:NMTOKEN">
                  <xsd:enumeration value="IN"/>
                  <xsd:enumeration value="OUT"/>
                  <xsd:enumeration value="INOUT"/>
               </xsd:restriction>
            </xsd:simpleType>
         </xsd:attribute>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="FormalParameters">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:FormalParameter" minOccurs="0"
maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="Icon" type="xsd:string"/>
   <xsd:element name="Implementation">
      <xsd:complexType>
         <xsd:choice>
            <xsd:element ref="xpdl:No"/>
            <xsd:element ref="xpdl:Tool" maxOccurs="unbounded"/>
            <xsd:element ref="xpdl:SubFlow"/>
         </xsd:choice>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="InitialValue" type="xsd:string"/>
   <xsd:element name="Join">
      <xsd:complexType>
         <xsd:attribute name="Type">
            <xsd:simpleType>
               <xsd:restriction base="xsd:NMTOKEN">
                  <xsd:enumeration value="AND"/>
```

```
                  <xsd:enumeration value="XOR"/>
               </xsd:restriction>
            </xsd:simpleType>
         </xsd:attribute>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="Length" type="xsd:string"/>
   <xsd:element name="Limit" type="xsd:string"/>
   <xsd:element name="ListType">
      <xsd:complexType>
         <xsd:group ref="xpdl:DataTypes"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="Manual">
      <xsd:complexType/>
   </xsd:element>
   <xsd:element name="Member">
      <xsd:complexType>
         <xsd:group ref="xpdl:DataTypes"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="No">
      <xsd:complexType/>
   </xsd:element>
   <xsd:element name="Package">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:PackageHeader"/>
            <xsd:element ref="xpdl:RedefinableHeader" minOccurs="0"/>
            <xsd:element ref="xpdl:ConformanceClass" minOccurs="0"/>
            <xsd:element ref="xpdl:Script" minOccurs="0"/>
            <xsd:element ref="xpdl:ExternalPackages" minOccurs="0"/>
            <xsd:element ref="xpdl:TypeDeclarations" minOccurs="0"/>
            <xsd:element ref="xpdl:Participants" minOccurs="0"/>
            <xsd:element ref="xpdl:Applications" minOccurs="0"/>
            <xsd:element ref="xpdl:DataFields" minOccurs="0"/>
            <xsd:element ref="xpdl:WorkflowProcesses" minOccurs="0"/>
            <xsd:element ref="xpdl:ExtendedAttributes" minOccurs="0"/>
         </xsd:sequence>
         <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
         <xsd:attribute name="Name" type="xsd:string"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="PackageHeader">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:XPDLVersion"/>
            <xsd:element ref="xpdl:Vendor"/>
            <xsd:element ref="xpdl:Created"/>
            <xsd:element ref="xpdl:Description" minOccurs="0"/>
            <xsd:element ref="xpdl:Documentation" minOccurs="0"/>
            <xsd:element ref="xpdl:PriorityUnit" minOccurs="0"/>
            <xsd:element ref="xpdl:CostUnit" minOccurs="0"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="Participant">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:ParticipantType"/>
            <xsd:element ref="xpdl:Description" minOccurs="0"/>
            <xsd:element ref="xpdl:ExternalReference" minOccurs="0"/>
            <xsd:element ref="xpdl:ExtendedAttributes" minOccurs="0"/>
         </xsd:sequence>
         <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
         <xsd:attribute name="Name" type="xsd:string"/>
      </xsd:complexType>
```



```
      </xsd:element>
      <xsd:element name="ParticipantType">
         <xsd:complexType>
            <xsd:attribute name="Type" use="required">
               <xsd:simpleType>
                  <xsd:restriction base="xsd:NMTOKEN">
                     <xsd:enumeration value="RESOURCE_SET"/>
                     <xsd:enumeration value="RESOURCE"/>
                     <xsd:enumeration value="ROLE"/>
                     <xsd:enumeration value="ORGANIZATIONAL_UNIT"/>
                     <xsd:enumeration value="HUMAN"/>
                     <xsd:enumeration value="SYSTEM"/>
                  </xsd:restriction>
               </xsd:simpleType>
            </xsd:attribute>
         </xsd:complexType>
      </xsd:element>
      <xsd:element name="Participants">
         <xsd:complexType>
            <xsd:sequence>
               <xsd:element ref="xpdl:Participant" minOccurs="0"
maxOccurs="unbounded"/>
            </xsd:sequence>
         </xsd:complexType>
      </xsd:element>
      <xsd:element name="Performer" type="xsd:string"/>
      <xsd:element name="Priority" type="xsd:string"/>
      <xsd:element name="PriorityUnit" type="xsd:string"/>
      <xsd:element name="ProcessHeader">
         <xsd:complexType>
            <xsd:sequence>
               <xsd:element ref="xpdl:Created" minOccurs="0"/>
               <xsd:element ref="xpdl:Description" minOccurs="0"/>
               <xsd:element ref="xpdl:Priority" minOccurs="0"/>
               <xsd:element ref="xpdl:Limit" minOccurs="0"/>
               <xsd:element ref="xpdl:ValidFrom" minOccurs="0"/>
               <xsd:element ref="xpdl:ValidTo" minOccurs="0"/>
               <xsd:element ref="xpdl:TimeEstimation" minOccurs="0"/>
            </xsd:sequence>
            <xsd:attribute name="DurationUnit">
               <xsd:simpleType>
                  <xsd:restriction base="xsd:NMTOKEN">
                     <xsd:enumeration value="Y"/>
                     <xsd:enumeration value="M"/>
                     <xsd:enumeration value="D"/>
                     <xsd:enumeration value="h"/>
                     <xsd:enumeration value="m"/>
                     <xsd:enumeration value="s"/>
                  </xsd:restriction>
               </xsd:simpleType>
            </xsd:attribute>
         </xsd:complexType>
      </xsd:element>
      <xsd:element name="RecordType">
         <xsd:complexType>
            <xsd:sequence>
               <xsd:element ref="xpdl:Member" maxOccurs="unbounded"/>
            </xsd:sequence>
         </xsd:complexType>
      </xsd:element>
      <xsd:element name="RedefinableHeader">
         <xsd:complexType>
            <xsd:sequence>
               <xsd:element ref="xpdl:Author" minOccurs="0"/>
               <xsd:element ref="xpdl:Version" minOccurs="0"/>
               <xsd:element ref="xpdl:Codepage" minOccurs="0"/>
               <xsd:element ref="xpdl:Countrykey" minOccurs="0"/>
```

```
            <xsd:element ref="xpdl:Responsibles" minOccurs="0"/>
         </xsd:sequence>
         <xsd:attribute name="PublicationStatus">
            <xsd:simpleType>
               <xsd:restriction base="xsd:NMTOKEN">
                  <xsd:enumeration value="UNDER_REVISION"/>
                  <xsd:enumeration value="RELEASED"/>
                  <xsd:enumeration value="UNDER_TEST"/>
               </xsd:restriction>
            </xsd:simpleType>
         </xsd:attribute>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="Responsible" type="xsd:string"/>
   <xsd:element name="Responsibles">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:Responsible" minOccurs="0"
maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="Route">
      <xsd:complexType/>
   </xsd:element>
   <xsd:element name="SchemaType">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="Script">
      <xsd:complexType>
         <xsd:attribute name="Type" type="xsd:string" use="required"/>
         <xsd:attribute name="Version" type="xsd:string" use="optional"/>
         <xsd:attribute name="Grammar" type="xsd:anyURI" use="optional"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="SimulationInformation">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:Cost"/>
            <xsd:element ref="xpdl:TimeEstimation"/>
         </xsd:sequence>
         <xsd:attribute name="Instantiation">
            <xsd:simpleType>
               <xsd:restriction base="xsd:NMTOKEN">
                  <xsd:enumeration value="ONCE"/>
                  <xsd:enumeration value="MULTIPLE"/>
               </xsd:restriction>
            </xsd:simpleType>
         </xsd:attribute>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="Split">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:TransitionRefs" minOccurs="0"/>
         </xsd:sequence>
         <xsd:attribute name="Type">
            <xsd:simpleType>
               <xsd:restriction base="xsd:NMTOKEN">
                  <xsd:enumeration value="AND"/>
                  <xsd:enumeration value="XOR"/>
               </xsd:restriction>
```



```
            </xsd:simpleType>
         </xsd:attribute>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="StartMode">
      <xsd:complexType>
         <xsd:choice>
            <xsd:element ref="xpdl:Automatic"/>
            <xsd:element ref="xpdl:Manual"/>
         </xsd:choice>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="SubFlow">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:ActualParameters" minOccurs="0"/>
         </xsd:sequence>
         <xsd:attribute name="Id" type="xsd:string" use="required"/>
         <xsd:attribute name="Execution">
            <xsd:simpleType>
               <xsd:restriction base="xsd:NMTOKEN">
                  <xsd:enumeration value="ASYNCHR"/>
                  <xsd:enumeration value="SYNCHR"/>
               </xsd:restriction>
            </xsd:simpleType>
         </xsd:attribute>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="TimeEstimation">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:WaitingTime" minOccurs="0"/>
            <xsd:element ref="xpdl:WorkingTime" minOccurs="0"/>
            <xsd:element ref="xpdl:Duration" minOccurs="0"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="Tool">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:ActualParameters" minOccurs="0"/>
            <xsd:element ref="xpdl:Description" minOccurs="0"/>
            <xsd:element ref="xpdl:ExtendedAttributes" minOccurs="0"/>
         </xsd:sequence>
         <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
         <xsd:attribute name="Type">
            <xsd:simpleType>
               <xsd:restriction base="xsd:NMTOKEN">
                  <xsd:enumeration value="APPLICATION"/>
                  <xsd:enumeration value="PROCEDURE"/>
               </xsd:restriction>
            </xsd:simpleType>
         </xsd:attribute>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="Transition">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:Condition" minOccurs="0"/>
            <xsd:element ref="xpdl:Description" minOccurs="0"/>
            <xsd:element ref="xpdl:ExtendedAttributes" minOccurs="0"/>
         </xsd:sequence>
         <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
         <xsd:attribute name="From" type="xsd:NMTOKEN" use="required"/>
         <xsd:attribute name="To" type="xsd:NMTOKEN" use="required"/>
         <xsd:attribute name="Name" type="xsd:string"/>
      </xsd:complexType>
```



```
   </xsd:element>
   <xsd:element name="TransitionRef">
      <xsd:complexType>
         <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="TransitionRefs">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:TransitionRef" minOccurs="0"
maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="TransitionRestriction">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:Join" minOccurs="0"/>
            <xsd:element ref="xpdl:Split" minOccurs="0"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="TransitionRestrictions">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:TransitionRestriction" minOccurs="0"
maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="Transitions">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:Transition" minOccurs="0"
maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="TypeDeclaration">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:group ref="xpdl:DataTypes"/>
            <xsd:element ref="xpdl:Description" minOccurs="0"/>
            <xsd:element ref="xpdl:ExtendedAttributes" minOccurs="0"/>
         </xsd:sequence>
         <xsd:attribute name="Id" type="xsd:ID" use="required"/>
         <xsd:attribute name="Name" type="xsd:string"/>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="TypeDeclarations">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:TypeDeclaration" minOccurs="0"
maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="UnionType">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element ref="xpdl:Member" maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:element name="ValidFrom" type="xsd:string"/>
   <xsd:element name="ValidTo" type="xsd:string"/>
   <xsd:element name="Vendor" type="xsd:string"/>
```

```
      <xsd:element name="Version" type="xsd:string"/>
      <xsd:element name="WaitingTime" type="xsd:string"/>
      <xsd:element name="WorkflowProcess">
         <xsd:complexType>
            <xsd:sequence>
               <xsd:element ref="xpdl:ProcessHeader"/>
               <xsd:element ref="xpdl:RedefinableHeader" minOccurs="0"/>
               <xsd:element ref="xpdl:FormalParameters" minOccurs="0"/>
               <xsd:element ref="xpdl:DataFields" minOccurs="0"/>
               <xsd:element ref="xpdl:Participants" minOccurs="0"/>
               <xsd:element ref="xpdl:Applications" minOccurs="0"/>
               <xsd:element ref="xpdl:ActivitySets" minOccurs="0"/>
               <xsd:element ref="xpdl:Activities" minOccurs="0"/>
               <xsd:element ref="xpdl:Transitions" minOccurs="0"/>
               <xsd:element ref="xpdl:ExtendedAttributes" minOccurs="0"/>
            </xsd:sequence>
            <xsd:attribute name="Id" type="xsd:NMTOKEN" use="required"/>
            <xsd:attribute name="Name" type="xsd:string"/>
            <xsd:attribute name="AccessLevel">
               <xsd:simpleType>
                  <xsd:restriction base="xsd:NMTOKEN">
                     <xsd:enumeration value="PUBLIC"/>
                     <xsd:enumeration value="PRIVATE"/>
                  </xsd:restriction>
               </xsd:simpleType>
            </xsd:attribute>
         </xsd:complexType>
      </xsd:element>
      <xsd:element name="WorkflowProcesses">
         <xsd:complexType>
            <xsd:sequence>
               <xsd:element ref="xpdl:WorkflowProcess" minOccurs="0"
   maxOccurs="unbounded"/>
            </xsd:sequence>
         </xsd:complexType>
      </xsd:element>
      <xsd:element name="WorkingTime" type="xsd:string"/>
      <xsd:element name="XPDLVersion" type="xsd:string"/>
      <xsd:element name="Xpression">
         <xsd:complexType mixed="true">
            <xsd:choice minOccurs="0" maxOccurs="unbounded">
               <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
            </xsd:choice>
         </xsd:complexType>
      </xsd:element>
   </xsd:schema>
```

10. Figures and Tables

10.1. Figures

Figure 5-1: The Concept of the Process Definition Interchange 7
Figure 6-1: Meta-Model top-level entities 8
Figure 6-3: Workflow Process Definition Meta Model 12
Figure 6-5: Package Definition Meta Model 13
Figure 7-1: Activity Structures & Transition Conditions 30
Figure 8-1: EOrder Main Process 53
Figure 8-2: CreditCheck Subprocess 53
Figure 8-3: FillOrder Subprocess 54

10.2. Tables

Table 6-1: Overview of Elements 15
Table 7-1: Extended Attributes -- Attributes 16
Table 7-3: Formal Parameters -- Attributes 17
Table 7-5: External Reference – Attributes 18
Table 7-7: Package Definition – Attributes 19
Table 7-9: Package Definition Header -- Attributes 20
Table 7-11: Redefinable Header -- Attributes 21
Table 7-13: Conformance Class Declaration -- Attributes 22
Table 7-15: Script – Attributes 23
Table 7-16: External Package Reference -- Attributes 23
Table 7-18: Workflow Application Declaration -- Attributes 24
Table 7-20: Workflow Process Definition – Attributes 25
Table 7-22: Workflow Process Definition Header -- Attributes 27
Table 7-24: Workflow Process Redefinable Header -- Attributes 28
Table 7-26: ActivitySet 29
Table 7-27: Entity type relationships for different Activity types 30
Table 7-29: Process Activity -- Attributes 31
Table 7-31: Execution Control -- Attributes 33
Table 7-33: Implementation Alternatives -- Attributes 33
Table 7-35: Tool -- Attributes 34
Table 7-37: Subflow -- Attributes 35
Table 7-39:Deadline 36
Table 7-40: Simulation Information – Attributes 38

Table 7-42: Transition Restrictions -- Attributes 38
Table 7-45: Join -- Attributes 39
Table 7-47: Split -- Attributes 40
Table 7-49: Transition Information -- Attributes 41
Table 7-51: Condition -- Attributes 42
Table 7-53: Workflow Participant -- Attributes 43
Table 7-55: Participant Entity Type – Attributes 44
Table 7-57: Workflow Relevant Data -- Attributes 45
Table 7-59: Standard Data Types 46
Table 7-61: Basic Data Types -- Attributes 47
Table 7-63: Record Type – Attributes 48
Table 7-65: Union Type 48
Table 7-67: Enumeration Type -- Attributes 49
Table 7-69: Array Type – Attributes 49
Table 7-71: List Type -- Attributes 49
Table 7-73: Type Declaration 50
Table 7-7438: Declared Data Type – Attributes 50

