Analysis Pipelines and Interoperability in the Morphometry Biomedical Informatics Research Network (mBIRN)

Shawn Murphy, Michael Mendis, David Kennedy, Jorge Jovicich, Randy Gollub, and Bruce Rosen

Abstract

The ability to send image data through a succession of software programs is critical for the successful analysis of complex imaging data. Applications that oversee this process are called "Workflow" engines, and their processing can be digested into machine-readable lists of instructions know as a "Workflow definition languages" (WDLs). Workflow engines of various kinds (Kepler, LONI, jBPM, TCL scripts) exist at the various mBIRN sites. Given the potential importance of workflow engines in performing image calculations, it would be of great interest to the mBIRN to adopt a common WDL language or WDL interconversion process that allows a workflow to be run at any of the mBIRN sites. The purpose of this report is to provide an understanding of the capabilities of the WDLs that are currently defined, and to understand the complexity of transformations that could take one WDL to another. In the context of mBIRN, if such transformations were possible, each site could run their own workflow engine, achieving interoperability by transforming one WDL into another. We found that the WDLs underlying the "scientific" workflow applications (MoML, XScufl, SWFL, possibly LONI pipeline) were similar in structure and are potentially interoperable, while the WDLs for business applications (BPEL4WS, WSFL, XPDL) are not. However, even the scientific WDLs could only be interconverted if there would exist a common repository of data type definitions and software modules such that they are available to perform the computations within the workflows.

Introduction

An analysis pipeline allows experiments to be calculated from beginning to end. As mBIRN resources have matured, the need to tie them together has evolved as an mBIRN-wide focus. The use of a well functioning analysis pipeline enables not only the initial calculation of the experimental results, but also the recalculation for verification of the results, and the exploration of the parameter space of the results. Without an efficient analysis pipeline, the assumptions embedded in an analysis cannot be tested and explored. A unique opportunity to explore the parameter space exists with radiological data which exists almost entirely in electronic format, that is, much of the experiment using radiological data is performed "in silico". Many of the initial parameters used for calculations are mostly guesses. These guesses may never be tested, especially in very complex calculations requiring many different programs to process the data. It is usually unknown how the entire calculation is dependent on these guesses. Each initial parameter used in a calculation will be associated with a certain influence on the results. If the parameter is changed slightly, the results may be the same, or the results may change greatly. The amount of change in the results per change of an initial parameter may be graphed as a result-dependency vector space, and this graph will show where care must be taken with the initial guesses. Constructing such a vector space may require many millions of recalculations, especially to explore the interactions of parameters. This will only be possible with well functioning analysis pipelines.

Focusing on analysis pipelines will entail unifying data processing pipeline solutions at various current mBIRN sites to become a single mBIRN solution. This solution is not envisioned to be exactly the same software running at every site, since legacy and local requirements may not make a direct software solution desirable. Rather, it is envisioned that as long as the workflow software is able to adhere to a short list of requirements that a single workflow definition language can function across various workflow engines. The benefits of a single mBIRN workflow definition language will be that collaborations between sites can be set up faster with the ability to routinely explore complex parameter spaces. Initial requirements for an mBIRN pipeline solution will be to build a completely open workflow definition language that is based upon existing workflow expression standards and ensure architecture can solve current mBIRN use-cases (SASHA, MIRIAD, and MAD).

Since analysis pipelines in science are roughly equivalent to workflow definitions in business, we will also be looking to business for workflow specifications. In business, as in science, workflow definition languages are used to embody the requirements for the workflow applications. Current business workflow languages that appear to embody some of our requirements are the XML Process Definition Language (XPDL, sponsored by the Workflow Management Coalition) and the Business Process Execution Language for Web Services (BPEL4WS, sponsored by a coalition of BEA, IBM, Microsoft, SAP AG and Siebel Systems). Scientific analysis pipeline languages that appear to embody some of our requirements include XML Simple Conceptual Unified Flow (XScufl, sponsored by the myGrid project), Modeling Markup Language (MoML, sponsored by the Kepler project), Service Workflow Language (SWFL, sponsored by GridLab project), and Laboratory of Neuro Imaging XML Workflow Representation (LONI-XML, sponsored by the Laboratory of Neuro Imaging).

At the March mBIRN all hands meeting, we discussed various functions that would be necessary for an adequate workflow engine. These functions included both requirements for control of the sequence of activities to be performed, as well as the functions themselves. The control of the workflow is determined largely by the workflow definition language, while the function of the workflow engine is determined largely by the programming modules that it supports. This paper focuses mostly on the control of the workflow and how to define it. The evolving architecture of the actual workflow application will be described in the section on future directions. However, the adoption of existing applications (such as the LONI analytic pipeline) as a starting point to build the architecture is highly desirable. The architecture later described would then be layered upon existing applications and evolve in an mBIRN open source setting.

Methods

We wished to understand what constituted the common elements of the various workflow definitions. In order to achieve this, we looked in detail at the following scientific workflow representations. All of the workflows were represented in XML. Detailed specifications of the representations are available in the appendixes. Descriptions of the scientific workflow applications that implement these specifications are included in appendix 1. Full specifications of these workflow definition languages are defined in the appendixes.

XML Simple Conceptual Unified Flow (XScufl)
http://taverna.sourceforge.net/docs/xscuflspecification.html

The XScufl workflow description language is used by the Taverna project to store and retrieve workflow definitions. The language itself should be considered volatile, and they do not support this language for use outside of Taverna. However, it contains the elements that appear to be most necessary in constructing scientific workflows.

Top level tag for an XScufl definition, contains arbitrary numbers of processor, link, source, sink and coordination elements defining member components of this workflow.

<scufl version(string) log(int)>
<processor> - Defines a single atomic processing operation within a workflow.
<link>* - the link transfers data from an output port to an input port, regarding workflow sources as output ports (from the workflow perspective they are providing data) and workflow sinks as inputs (they receive data)
<source>* - used to declare top level workflow inputs and outputs
<sink>* - used to declare top level workflow inputs and outputs
<coordination>* - Coordination constraints are used to prevent a processor transitioning between states until some constraint condition has been satisfied.

Service Workflow Language (SWFL) (Triana implements a subset)
http://www.cs.cf.ac.uk/User/Yan.Huang/GridWF/SWFL.htm

SWFL is an XML-based Meta language for the construction of scientific workflows from OGSA-compliant services. SWFL was developed at Cardiff University in 2002 and 2003. SWFL extends IBM's WSFL and supports a new set of conditional operators and loop constructs as well as arrays and objects.

In WSFLs flow metamodel, an activity represents a task to be performed as a single step and is implemented by a Web service. If the model described by a WSFL document is viewed as a directed graph then the activities correspond to the nodes of the graph. Activities are wired together through control links that represent the control flow of the flow model. A control link is a directed edge that prescribes the order in which the activities must be performed. A data link is a second kind of directed edge in the graph structure and specifies the flow of data between a source activity and a target activity. For a given data link, a *map* describes how a field in the message part of the target's input message is constructed from a field of the source's output message. A flow model itself also has input and output: a *flow source* is a source of the data links targeting activities in the flow model, and a *flow sink* is a target of the data links out of the flow model

A new element is introduced into the schema called jactivityType that extends the activity type to six kinds of activities: normal, if, while, dowhile, for, and switch. The normal activity is the same as an activity defined in WSFL and is of type wsfl:activityType. The others are newly-

defined activity types corresponding to the conditional and loop constructs in the Java programming language. The for, while, and dowhile activities are all of type loopType, and the if and switch activities are of type controlType.

Modeling Markup Language (MoML) (Kepler)
http://ptolemy.eecs.berkeley.edu/

Ptolemy leverages an XML-meta language called MoML to produce a workflow document describing the relationships of the entities, properties, and ports in a workflow. Presently, Ptolemy actor libraries exist for the domains of bioinformatics and ecology at NCSU and SDSC.

Modeling Markup Language [1] (MoML) is an XML meta-language primarily used for expressing models of software systems. MoML defines tags for Entities, Properties, Relations, Links, and Ports, as decoupled from a specific class or service that the model represents.

MoML could be used to model a software program where each class is an entity, and each method is a collection of input and output ports. Data fields are modeled as properties. Entities can be inherited from each other and linked to each other.

A MoML document could also be used to define an abstract web service composition workflow. Each entity is a service provider, and services have a collection of ports. There is nothing in MoML that would prevent one from using it to model a grid services composition workflow.

LONI Pipeline
http://www.loni.ucla.edu/twiki/bin/view/Pipeline/

At the top of every pipeline module file is a set of class definitions. These class definitions define the various classes of objects to be manipulated, e.g. File, Function, Pipeline. They can denote inheritance relations. Once the classes have been defined, class instances are defined. All objects are resources, and provided an ID, to which it can be referred. Hence functions are defined as resources as well.

Everything in a pipeline is a resource. Every resource can be uniquely identified by its ID, specified by its "rdf:ID" attribute. IDs are not meant to be human readable. In order to define relationships between resources, there is a manner of referring to any resource from any other resource. This is done by using the attribute "rdf:resource". Every resource can have a set of labels. These labels are intended to be short, human readable names that identify the resource to a human. Every resource can have a set of comments. These comments are intended to provide (possibly verbose) explanations of the resources themselves.

XPDL
http://www.wfmc.org/standards/XPDL.htm

The XPDL language is structured around the 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.).

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 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.

BPEL4WS
http://www-128.ibm.com/developerworks/library/specification/ws-bpel/

BPEL4WS defines a model and a grammar for describing the behavior of a business process based on interactions between the process and its partners. The interaction with each partner occurs through Web Service interfaces, and the structure of the relationship at the interface level is encapsulated in what is called a partner link. The BPEL4WS process defines how multiple service interactions with these partners are coordinated to achieve a business goal, as well as the state and the logic necessary for this coordination. BPEL4WS also introduces systematic mechanisms for dealing with business exceptions and processing faults. Finally, BPEL4WS introduces a mechanism to define how individual or composite activities within a process are to be compensated in cases where exceptions occur or a partner requests reversal.

BPEL4WS is layered on top of several XML specifications: WSDL 1.1, XML Schema 1.0, and XPath1.0. WSDL messages and XML Schema type definitions provide the data model used by BPEL4WS processes. XPath provides support for data manipulation. All external resources and partners are represented as WSDL services. BPEL4WS provides extensibility to accommodate

future versions of these standards, specifically the XPath and related standards used in XML computation.

Analysis

It was our goal to not create new workflow definition languages, but rather to understand what was currently useful by analyzing currently operative WDLs. This allowed us to focus on control patterns that have been proven useful such that use cases have driven them to be implemented. This also allows us to achieve the necessary result of our paper, which is to understand what is required for an mBIRN workflow definition language to support if it is to interconvert into some of these currently implemented workflow definition languages.

We analyzed workflow representations in two ways. First, we explored the XML representations and attempted to understand from the representations themselves what the capabilities of those representations were. This had already been undertaken for business workflows [8, 9], and we followed a similar strategy to investigate scientific workflows. The results are detailed below, and more details about the methods behind this analysis are available at http://is.tm.tue.nl/research/patterns/patterns.htm
.

Basic Control Patterns
Sequence - execute activities in sequence
Parallel Split - execute activities in parallel
Synchronization - synchronize two parallel threads of execution
Exclusive Choice - choose one execution path from many alternatives
Simple Merge - merge two alternative execution paths
Advanced Branching and Synchronization Patterns
Multiple Choice - choose several execution paths from many alternatives
Synchronizing Merge - merge many execution paths. Synchronize if many paths are taken. Simple merge if only one execution path is taken
Multiple Merge - merge many execution paths without synchronizing
Discriminator - merge many execution paths without synchronizing. Execute the subsequent activity only once
N-out-of-M Join - merge many execution paths. Perform partial synchronization and execute subsequent activity only once
Structural Patterns
Arbitrary Cycles - execute workflow graph w/out any structural restriction on loops
Implicit Termination - terminate if there is nothing to be done
Patterns Involving Multiple Instances
MI without synchronization - generate many instances of one activity without synchronizing them afterwards
MI with a priori known design time knowledge - generate many instances of one activity when the number of instances is known at the design time (with synchronization)

MI with a priori known runtime knowledge - generate many instances of one activity when a number of instances can be determined at some point during the runtime (as in FOR loop but in parallel)
MI with no a priori runtime knowledge - generate many instances of one activity when a number of instances cannot be determined (as in WHILE loop but in parallel)
State-based patterns
Deferred Choice - execute one of the two alternatives threads. The choice which thread is to be executed should be implicit.
Interleaved Parallel Routing - execute two activities in random order, but not in parallel.
Milestone - enable an activity until a milestone is reached
Cancellation Patterns
Cancel Activity - cancel (disable) an enabled activity
Cancel Case - cancel (disable) the process
Sub-workflow – place a workflow within a larger workflow

pattern	XPDL	BPEL	WSFL	MoML	Scufl	SWFL	LONI
Sequence	+	+	+	+	+	+	+
Parallel Split	+	+	+	+	+	+	+?
Synchronization	+	+	+	+	+	+	+?
Exclusive Choice	+	+	+	+	+	+	?
Simple Merge	+	+	+	+	+	+	?
Multi Choice	+	+	+	+/-	-	+	?
Synchronizing Merge	+	+	+	+/-	-	+	?
Multi Merge	-	-	-	-	-	-	?
Discriminator	-	-	-	-	-	-	?
Arbitrary Cycles	+	-	-	+	-	+	?
Implicit Termination	+	+	+	-	-	-	?
MI without Synchronization	+	+	+	-	-	-	-
MI with a Priori Design Time Knowledge	+	+	+	-	-	-	-
MI with a Priori Runtime Knowledge	-	-	-	-	-	-	-
MI without a Priori Runtime Knowledge	-	-	-	-	-	-	-
Deferred Choice	-	+	-	-	-	-	-

Interleaved Parallel Routing	-	+/-	-	+/-	-	+/-	?
Milestone	-	-	-	+	-	-	-
Cancel Activity	-	+	+	-	-	-	?
Cancel Case	-	+	+	+	+	+	?
Sub-workflow	+	+	+	+	+	+	?

From the above analysis, it appears that the patterns that are supported by most scientific workflows are Sequence (execute activities in sequence), Parallel Split (execute activities in parallel), Synchronization (synchronize two parallel threads of execution), Exclusive Choice (choose one execution path from many alternatives), Simple Merge (merge two alternative execution paths), Arbitrary Cycles (execute workflow graph w/out any structural restriction on loops), Cancel Case (cancel the process), and Sub-workflow (place a workflow within a larger workflow). The business WDLs are much richer in control structures. Note that we were unable to analyze the LONI WDL capabilities in their entirety as we only had a limited specification. However, it generally appeared to follow the pattern capabilities of the scientific workflows.

Other differences were also noted between the scientific and the business workflows. As can be seen in the XPDL and BPEL4WS workflow representations included in the appendixes, the business workflow representations are oriented around business transactions. These transactions are represented at an extremely atomic level. The scientific workflows tended to be more abstract, and much of their capabilities may be achieved only by the functionality provided by programming modules that exist in the workflow. This became clear when comparing MoML and SWFL.

The next task in our analysis was to gain some insight into the basic structure of the workflow languages in order to understand their similarities and differences. This would allow us to further understand if transformations were possible between the WDLs. When we began our comparison, it became immediately apparent that the business workflow languages were much more complex than the scientific workflow languages, and had a structure considerably different that the scientific workflow languages. It seemed that any comparison between the two would be forced and unfruitful, therefore we focused on comparing the scientific workflow engines.

To these ends, we downloaded all 4 workflow reference implementations (LONI pipeline V3, Triana, Taverna, and Kepler. We attempted to represent a common workflow across all 4 scientific workflow applications to make our comparisons easier. This was not intended to be a "test" of the implementations, because implementation details (such as what modules are available) prevented this workflow from being successfully implemented in LONI and Triana. However, by picking a common and relatively simple workflow we could compare specifics across the WDLs.

The workflow we chose was to go to an internet URL and display an image from the html page at that URL. Specifically, the WDL was to direct the workflow engine to go to the http://www.dilbert.com, get the cartoon of the day, and display the image. The results are shown below for XScufl (Taverna) and MoML (Kepler). The results of the other two workflows are

included in the appendix. The full representations are included in the appendix, and can be run successfully after downloading the workflow applications.

```
<?xml version="1.0" encoding="UTF-8" ?>
- <s:scufl xmlns:s="http://org.embl.ebi.escience/xscufl/0.1alpha" version="0.2" log="0">
    <s:workflowdescription lsid="urn:lsid:www.mygrid.org.uk:operation:VI9FMF5HBQ3"
      author="Tom Oinn" title="Fetch today's Dilbert comic">Use the local java plugins and
      some filtering operations to fetch the comic strip image from
      http://www.dilbert.com</s:workflowdescription>
  - <s:processor name="dilbertURL">
      <s:stringconstant>http://www.dilbert.com/</s:stringconstant>
    </s:processor>
  - <s:processor name="getPage">
      <s:local>org.embl.ebi.escience.scuflworkers.java.WebPageFetcher</s:local>
    </s:processor>
  - <s:processor name="getComicStrip">
      <s:local>org.embl.ebi.escience.scuflworkers.java.WebImageFetcher</s:local>
    </s:processor>
  - <s:processor name="comicURLRegex">
      <s:stringconstant>.*/archive/images/dilbert.*</s:stringconstant>
    </s:processor>
  - <s:processor name="findComicURL">
      <s:local>org.embl.ebi.escience.scuflworkers.java.FilterStringList</s:local>
    </s:processor>
  - <s:processor name="getImageLinks">
      <s:local>org.embl.ebi.escience.scuflworkers.java.ExtractImageLinks</s:local>
    </s:processor>
    <s:link source="dilbertURL:value" sink="getPage:url" />
    <s:link source="getPage:contents" sink="getImageLinks:document" />
    <s:link source="getImageLinks:imagelinks" sink="findComicURL:stringlist" />
    <s:link source="comicURLRegex:value" sink="findComicURL:regex" />
    <s:link source="dilbertURL:value" sink="getComicStrip:base" />
    <s:link source="findComicURL:filteredlist" sink="getComicStrip:url" />
    <s:link source="getComicStrip:image" sink="todaysDilbert" />
  - <s:sink name="todaysDilbert">
    - <s:metadata>
      - <s:mimeTypes>
          <s:mimeType>image/*</s:mimeType>
        </s:mimeTypes>
      </s:metadata>
    </s:sink>
  </s:scufl>
```

HEADER

ACTORS

LINKS

We found that the scientific WDLs MoML, XScufl., and SWDL are generally divided into sections which are roughly Header information, Module/Actor information, Link information, and Synchronization (between modules) information. Unfortunately, because of the simplicity of the representation chosen, it did not show the Synchronization information so I will simply describe it below.

Header information sets up the workflow environment, often pointing to external applications that will be necessary and other requirements. Module information defines specifics linking the external applications and internal components to runnable instances in the workflow. Link information specifies how data is input and output from each runnable instance in the workflow. It usually will specify the data type, from what instance, and to what instance.

Synchronization information was more unique to each workflow. Synchronization information generally directs that one program module must finish before another starts, even though the programming modules may not be in direct series. Modeling Markup Language was somewhat unique in that it had the concept of a director. This is a way to specify the details of data flow using an overseeing application. SWFL had the most expressive control structures, but the reference implementation (Triana) did not use most of them. This reference implementation

appeared to be at an early stage of development, and indeed we could not get the reference workflow to work with it.

Discussion

It appears from our analysis that the workflow definition languages (WDLs) may be roughly divided into scientific and business definition languages (although SWFL lies somewhat in the middle). The scientific WDLs are less complex than the business WDLs and achieve most of their functionality through the program modules present within them. The business workflows are more sophisticated, and contain more control language, for example the ability to generate many instances of one activity without synchronizing them afterwards. In general, the business workflows are geared toward a complex transaction flow of data that is consumed by relatively short running processes. The business WDLs are specified to a much greater degree than the Scientific WDLs, and in general these languages may be considered a superset of the Scientific WDLs.

All of the WDLs allow for a good deal of customization for unforeseen needs. Definitions of data types are extremely critical for scientific workflows, because the data types tend to be varied (such as many different image formats, and many different versions of these formats). In general the business workflows are relatively weak at expressing data type. However, this issue is also not handled well in scientific workflows. Although many data types are allowed, classification of these types are arbitrary and not standardized, such that many workflow constructions would fail because of data type mismatches.

The ability to insert new application modules is also critical for scientific workflows. Of note, long running workflows have not been well supported in the business workflows. The scientific workflow implementations tend to be better at running workflows that go on for days, although the WDLs of business workflows would have no trouble handling these in principle. The business workflows have much more built in support for error handling, coordinating between application modules that are running, and stopping a sequence pending human intervention.

The primary conclusion drawn from this work is that the representation of scientific workflows in considerably different from business workflows. Having so stated, it is true that the business workflows are essentially Turing complete. Therefore, they could be manipulated to be used to represent scientific workflow. However, if one was to implement the full specification of one of the business workflows just to achieve a scientific workflow, a considerable amount of effort would need to be expended on parts of the specification that would not be needed for science. Furthermore, the representation would likely be cumbersome and difficult to maintain.

The secondary conclusion drawn from this work is that the representations of two of the scientific workflow representations, MoML and XScufl, are similar and potentially interoperable. The LONI specification has many similarities (see appendix), but could not be properly evaluated without the source code and further documentation. We anticipate being able to undertake a more complete evaluation of the interoperability of the LONI pipeline workflow representation in the near future. The SWFL representation is much more complex that MoML or XScufl. However, using the reference representation as a guide (Triana), it appears that this is

perhaps not the best fit for scientific workflows. In fact, the workflow application that is was targeted to implement SWFL, Triana, only implemented a small fraction of the SWFL specification, and that fraction (see appendix) looks very similar to what is implemented in MoML and XScufl. Therefore, it appears to confirm the suspicion that only a limited subset of control structures is truly necessary to support scientific workflows.

Finally, it is apparent that it will not be sufficient for simple transformations between WDLs to make them interoperable. In order for scientific workflows to be interoperable, it is critical for there to be a repository of application modules and data types. Web services may help to achieve the need for an application module repository in a transparent fashion. However, data types and transformations between them need to be rigorously specified.

There is a more subtle difficulty it the specification of the scientific workflows. The business workflows often have the available enumerated options build into the specification. However, the scientific workflows assume prior knowledge of this list, an assumption which will not allow them to be interoperable.

Both Scientific and Business workflows rely upon a repository of names for physical entities. For example, when a UPC symbol is passed in a business workflow, it is assumed that databases exist to look it up. Scientific workflows are relying on ad-hoc names for these entities, and this will not be interoperable in different settings.

Therefore, we would define the next steps as follows:

1) Arrive at a consensus for a set of workflow languages that can at be converted using a specific transformation sequence. This should be possible by adhering to the support of a specific set of workflow patterns. Guaranteeing the existence of appropriate modules and data types will be addressed in (2) below. Importantly, the workflow engines of mBIRN should be able to support these patterns, but more importantly, workflows should not be expected to support patterns outside of this set. From the analysis, it appears that the patterns that are supported by most scientific workflows are Sequence (execute activities in sequence), Parallel Split (execute activities in parallel), Synchronization (synchronize two parallel threads of execution), Exclusive Choice (choose one execution path from many alternatives), Simple Merge (merge two alternative execution paths), Arbitrary Cycles (execute workflow graph w/out any structural restriction on loops), Cancel Case (cancel the process), and Sub-workflow (place a workflow within a larger workflow). Clearly the business workflows are capable of many more patterns than these, although if they were somehow artificially restricted to this set of patterns they could also potentially be converted with the scientific workflows. Complex activity requiring patterns outside of these specifications could possibly be supported if specific modules existed for the engines to support them.

2) Set up a repository of programming modules, web-services, and data types to be used in workflows. The scientific workflows achieve most of their functionality through the software programs that they tie together. These programs are highly specialized and rapidly changing. If one expects scientific workflow programs to operate with these programs, they must be available. The use of web services to conduct these special functions takes away the challenge

of delivering these programs to the workflow engines environment, but the locations of the web services will still need to be published. Finally, the data types used by the programs can be complex and specific down to sub-versions of format types. A repository of data type definitions (commonly know as “standards”) and conversion programs that allow one data type to be converted to another will be necessary to achieve workflow engine convertibility.

3) Set up services to provide enumerated values to be used as choices in the workflows. Workflow definition languages provide an organization of slots in a document, similarly to providing a standard request form to fill out. It does not provide an interchangeable language to be used to fill out the form. For example, the programming modules and data types referred to in (2) will need to have names that are entered in the workflow diagram. Either these names will need to be presented prospectively to the users while they are creating their workflows, or a retrospective mapping will need to be made between names to keep them organized and pointing to the same module. This service will also allow mappings to be expressed between physical entities that have different names at different sites or in different workflows but refer to the same object.

References

[1] Edward A. Lee and Steve Neuendorffer. MoML — A Modeling Markup Language in XML — Version 0.4. Technical report, University of California at Berkeley, March, 2000. Online. Available:
http://www.gigascale.org/pubs/16/moml_erl_memo.pdf

[2] Scientific Data Management Center at NC State. 2003. Online. Available:
http://sdm.csc.ncsu.edu/

[3] Science Environment for Ecological Knowledge. 2003. Online. Available:
http://seek.ecoinformatics.org/

[4] Ilkay Altintas, Sangeeta Bhagwanani, David Buttler, Sandeep Chandra, Zhengang Cheng, Matthew A. Coleman, Terence Critchlow, Amarnath Gupta, Wei Han, Ling Liu, Bertram Ludascher, Calton Pu, Reagan Moore, Arie Shoshani, Mladen Vouk, "A Modeling and Execution Environment for Distributed Scientific Workflows" to be published in "Real World Semantic Web Applications", IOS Press, editor V. Kashyap, 2002. [Online]. Available:
http://renoir.csc.ncsu.edu/Faculty/Vouk/Papers/Cheng/BookChapter/Cheng_IOS_03.pdf

[5] Ewa Deelman, James Blythe, Yolanda Gil, and Carl Kesselman, "Workflow Management in GriPhyN," Chapter in “Grid Resource Management,” J. Nabrzyski, J. Schopf, and J. Weglarz editors, Kluwer, 2003.
http://www.isi.edu/~deelman/Pegasus/grm_chapter.pdf

[6] B. Ludaescher, A. Gupta, and M. E. Martone, "A Model-Based Mediator System for Scientific Data Management", bibl. Morgan Kaufmann, Chapter in: "Bioinformatics: Managing Scientific Data," editors, T. Critchlow and Z. Lacroix, 2003.
http://citeseer.nj.nec.com/cache/papers/cs/27492/http:zSzzSzwww.sdsc.eduzSz~guptazSzpublica

tionszSzmbm-chapter-rev.pdf/a-model-based-mediator.pdf

[7] Sandeep Chandra, "Service-based Support for Scientific Workflows" Thesis, 2002.[Online]. Available:
http://renoir.csc.ncsu.edu/Faculty/Vouk/Papers/Chandra/Chandra_MS_Thesis.pdf

[8] W.M.P. van der Aalst, A.H.M. ter Hofstede, B. Kiepuszewski, and A.P. Barros. Workflow Patterns. Distributed and Parallel Databases, 14(3), pages 5-51, July 2003.

.
[9] W.M.P. van der Aalst. Patterns and XPDL: A Critical Evaluation of the XML Process Definition Language. (PDF, 223 Kb). QUT Technical report, FIT-TR-2003-06, Queensland University of Technology, Brisbane, 2003.

Appendixes

Scientific Workflow Applications

LONI Pipeline
http://www.loni.ucla.edu/twiki/bin/view/Pipeline/

The LONI Pipeline is a simple graphical environment for constructing complex scientific analyses of data. It provides a visually intuitive interface to data analysis while also allowing for diverse programs to interact seamlessly. The Pipeline allows researchers to share their methods of analysis with each other easily and provides a simple platform for distributing new programs, as well as program updates, to the desired community. The environment also takes advantage of supercomputing environments by automatically parallelizing data-independent programs in a given analysis whenever possible.

A modular architecture, with task specific components that provide quick and easy customization, is currently under development. Taking advantage of this has allowed us to make other dramatic improvements (which are included in the upcoming v3 release). These new improvements include improved security, grid computing integration, automated process management, automated resource management, advanced debugging, and fault tolerance.

XML Simple Conceptual Unified Flow (XScufl) (Taverna)
http://www.mygrid.org.uk

The myGrid Project is a collection of bioinformatics web services and grid services hosted by the European Bioinformatics Institute. myGrid uses SoapLab and the Apache Axis framework to provide a web service interface to a collection of grid-based data-analysis services. Scientists are able to compose, edit, and save workflows with either a web portal or a GUI workbench. The system currently supports two XML workflow languages, a subset of IBM's WSFL and a more domain-specific language called XScufl. myGrid executes workflows defined in these documents with the IT Innovation Workflow Enactment Engine.

Service Workflow Language (SWFL) (Triana implements a subset)

http://www.triana.co.uk/

Triana is a set of open source java libraries that provide a GUI interface for building workflows from a collection of OGSA grid services. Triana has been leveraged by several other projects including myGrid and Chimera. Triana contains an engine for coordinating and invoking a set of grid services. Triana also contains a peer-to-peer component based on JXTA allowing Triana to run a variety of devices including PDAs and mobile phones.

Modeling Markup Language (MoML) (Kepler)
http://ptolemy.eecs.berkeley.edu/

Ptolemy-II is a visual modeling tool written in Java. It was begun in 1997 at UC Berkley. Several recent SDM efforts have extended the Ptolemy-II platform to allow for the drag-and-drop creation of scientific workflows from libraries of actors. The Ptolemy actor is often a wrapper around a call to web service or grid service. Ptolemy leverages an XML-meta language called MoML to produce a workflow document describing the relationships of the entities, properties, and ports in a workflow. Presently, Ptolemy actor libraries exist for the domains of bioinformatics and ecology at NCSU and SDSC.

The process of creating a workflow with the Ptolemy software is centered on creating Java classes that extends a build-in Actor class. Usually, the Actor corresponds to an MoML entity, and the Actors is a wrapper around a web service stub or a local program. A concrete workflow consists of a series of these actors interacting in way deemed legal by the abstract workflow document.

This approach to workflow construction is discussed in [6] and [7].

BioPipe
http://www.biopipe.org/

BioPipe is one of a series of related packages for carrying out bioinformatics analysis from the Open Bioinformatics Foundation. Biopipe is a collection of Perl modules for constructing workflows from BioPerl applications. Much of the code and ideas are borrowed from the Ensembl pipeline project.

The GriPhyN project (No current web site)
http://www.isi.edu/~deelman/Pegasus/grm_chapter.pdf

National Science Foundation-Funded Grid Physics Network (GriPhyN). This project leverages a collection of grid technologies including Condor-G, DAG-Man, and Globus.

The workflow management software in GriPhyN is used to ensure that the grid can provide certain ‘data products’ to its users. As in the Ptolemy system, there are a set of pre-set components which can be used to form a workflow. Specifically, the system is concerned with translating user requests to the grid into a series of job submissions and data requests.

The abstract workflow is used as a planning mechanism, because when the abstract workflow is defined it is checked again the actual available services.

Selecting and configuring application components to form an abstract workflow. The application components are selected by examining the specification of their capabilities and checking to see

if they can generate the desired data products (p. 2).

It seems that the abstract workflow is represented as a data structure in the Pegasus middleware. The concrete workflow is expressed as a multi-part job.

SCIRUN
http://software.sci.utah.edu/scirun.html

SCIRUN is an application begun in 1992 by National Center for Research Resources (NCRR) Center at Utah. SCIRUN is a GUI "scientific workbench" that allows users to construct, manage, and debug simulations in domains such as physics and neurobiology. SCIRUN simulation may be thought of as workflows since they allow for parallel and conditional execution of tasks. Some SCIRUN applications allow for jobs to be run on a grid. SCIRUN also provides extensive scientific visualization libraries.

WASA
http://dbms.uni-muenster.de/menu.php3?item=projects&page='wasa/index.php3?id=1'

WASA2 is an application that supports the creation and execution of workflows from CORBA components. It includes a GUI workflow modeler as well as controls to modify a workflow while it is running. WASA2 has been used in the domains of geoprocessing and molecular biology as well as for modeling business processes. WASA2 uses a strictly object orient approach where workflows are represented as CORBA objects and can be displayed as UML diagrams.

Chimera
http://www.griphyn.org/chimera/

Chimera a system used to find or create a workflow for a series of OGSA grid services to provide a scientist's requested "data product." Chimera was begun in 1999 and is enabled by the Pegasus Planner, a GriPhyN project at ISI. The Chimera is middleware designed to be invisible from a client who requests the data product. The workflow is represented as an "abstract program execution graph." This graph is transformed into an executable DAG for the Condor DAGman scheduler.

DiscoveryNet
http://www.discovery-on-the.net

DiscoveryNet is a collection of software built on top of the UNICORE grid system for arranging database access and knowledge discovery procedures. DiscoveryNet was begun in 2001 at Imperial College of Science. DiscoveryNet provides a means of describing workflow between analysis service providers, data owners, and scientists who arrange and execute these workflows. DiscoveryNet makes use of the OSGSI components and protocols as well as its own protocol for workflows, Discovery Process Markup Language (DPML). This language is used for constructing, running, and managing grid-services, as well as recording their history

wftk
http://www.vivtek.com/wftk/

Open-source workflow toolkit, or wftk, is the name of a generalized workflow system, implemented as a series of Java libraries. wftk was begun in 1998 by Michael Roberts. wftk uses its own high level language to describe workflow, and stores workflow related documents in a series of XML "datasheets." wftk workflow engine contain two models of a workflow: a "task-based" model, essentially a DAG, and a "state-based" model, essentially a FSM. wftk libraries can also be run as web services.

ICENI
http://www.lesc.ic.ac.uk/iceni/

ICENI (Imperial College e-Science Net-worked Infrastructure) is a collection of grid middleware used for providing and coordinating grid services for e-science applications. ICENI includes a GUI workflow construction tool integrated into the NetBeans IDE. This tool can create a textual "execution plan" of the workflow in an XML-meta language derived from YAWL (Yet Another Workflow Language). The ICENI workflow system supports conditionals, loops, and parallel execution.

BioOpera
http://www.inf.ethz.ch/personal/bausch/bioopera/main.html

BioOpera is an application for the composition of various bioinformatics applications built on top of the OPERA architecture. The project was begun in 1999 at the Swiss Federal Institute of Technology. BioOpera provides a GUI tool for the construction of workflows from web services and grid services. BioOpera represents a workflow "process template" as a DAG, and translates this set of activities into an execution script for Condor-G. Additionally, the web service invocation engine within BioOpera uses UDDI, and the grid service component uses service description documents.

ILab
http://www.fhrg.fhg.de/index_en.html

ILab is a set of grid middleware developed at The Fraunhofer ICT Group since 2001. ILab includes a GUI tool for the construction of workflows from grid services. This tool uses an XML-based language called Grid Application Definition Language (GADL) to assemble applications from grid services, and Grid Job Definition Language (GJobDL) to describe the runtime behavior of such applications. ILab uses Petri nets instead of DAGs to model and control the workflow.

GridAnt
http://www-unix.globus.org/cog/projects/gridant/

GridAnt is an extension of the Apache Ant build tool residing in the Globus COG kit. GridAnt

allows for the construction of client side workflow for Goblus Toolkit 3 application by allows for the specification of precondition and parallel tasks in much the same way as the Ant build tool.

XScufl Analysis

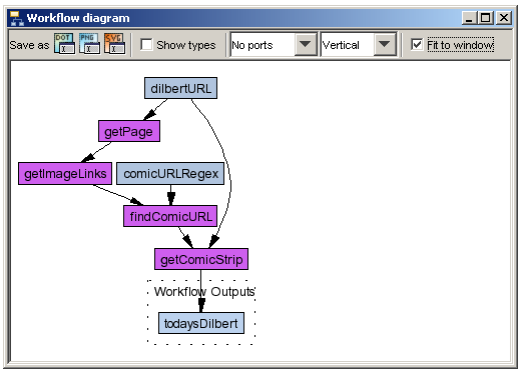
```
<?xml version="1.0" encoding="UTF-8" ?>
- <s:scufl xmlns:s="http://org.embl.ebi.escience/xscufl/0.1alpha" version="0.2" log="0">
    <s:workflowdescription lsid="urn:lsid:www.mygrid.org.uk:operation:VI9FMF5HBQ3"
      author="Tom Oinn" title="Fetch today's Dilbert comic">Use the local java plugins and
      some filtering operations to fetch the comic strip image from
      http://www.dilbert.com</s:workflowdescription>
  - <s:processor name="dilbertURL">
      <s:stringconstant>http://www.dilbert.com/</s:stringconstant>
    </s:processor>
  - <s:processor name="getPage">
      <s:local>org.embl.ebi.escience.scuflworkers.java.WebPageFetcher</s:local>
    </s:processor>
  - <s:processor name="getComicStrip">
      <s:local>org.embl.ebi.escience.scuflworkers.java.WebImageFetcher</s:local>
    </s:processor>
  - <s:processor name="comicURLRegex">
      <s:stringconstant>.*/archive/images/dilbert.*</s:stringconstant>
    </s:processor>
  - <s:processor name="findComicURL">
      <s:local>org.embl.ebi.escience.scuflworkers.java.FilterStringList</s:local>
    </s:processor>
  - <s:processor name="getImageLinks">
      <s:local>org.embl.ebi.escience.scuflworkers.java.ExtractImageLinks</s:local>
    </s:processor>
    <s:link source="dilbertURL:value" sink="getPage:url" />
    <s:link source="getPage:contents" sink="getImageLinks:document" />
    <s:link source="getImageLinks:imagelinks" sink="findComicURL:stringlist" />
    <s:link source="comicURLRegex:value" sink="findComicURL:regex" />
    <s:link source="dilbertURL:value" sink="getComicStrip:base" />
    <s:link source="findComicURL:filteredlist" sink="getComicStrip:url" />
    <s:link source="getComicStrip:image" sink="todaysDilbert" />
  - <s:sink name="todaysDilbert">
    - <s:metadata>
      - <s:mimeTypes>
          <s:mimeType>image/*</s:mimeType>
        </s:mimeTypes>
      </s:metadata>
    </s:sink>
</s:scufl>
```

HEADER

ACTORS

LINKS

Taverna Diagram



XScufl Dilbert Workflow

```
<?xml version="1.0" encoding="UTF-8"?>
<s:scufl xmlns:s="http://org.embl.ebi.escience/xscufl/0.1alpha" version="0.2"
log="0">
  <s:workflowdescription
lsid="urn:lsid:www.mygrid.org.uk:operation:VI9FMF5HBQ3" author="Tom Oinn"
title="Fetch today's Dilbert comic">Use the local java plugins and some
filtering operations to fetch the comic strip image from
http://www.dilbert.com</s:workflowdescription>
  <s:processor name="dilbertURL">
    <s:stringconstant>http://www.dilbert.com/</s:stringconstant>
  </s:processor>
  <s:processor name="getPage">
    <s:local>org.embl.ebi.escience.scuflworkers.java.WebPageFetcher</s:local>
  </s:processor>
  <s:processor name="getComicStrip">

<s:local>org.embl.ebi.escience.scuflworkers.java.WebImageFetcher</s:local>
  </s:processor>
  <s:processor name="comicURLRegex">
    <s:stringconstant>.*/archive/images/dilbert.*</s:stringconstant>
  </s:processor>
  <s:processor name="findComicURL">

<s:local>org.embl.ebi.escience.scuflworkers.java.FilterStringList</s:local>
  </s:processor>
  <s:processor name="getImageLinks">

<s:local>org.embl.ebi.escience.scuflworkers.java.ExtractImageLinks</s:local>
  </s:processor>
```

```
  <s:link source="dilbertURL:value" sink="getPage:url" />
  <s:link source="getPage:contents" sink="getImageLinks:document" />
  <s:link source="getImageLinks:imagelinks" sink="findComicURL:stringlist" />
  <s:link source="comicURLRegex:value" sink="findComicURL:regex" />
  <s:link source="dilbertURL:value" sink="getComicStrip:base" />
  <s:link source="findComicURL:filteredlist" sink="getComicStrip:url" />
  <s:link source="getComicStrip:image" sink="todaysDilbert" />
  <s:sink name="todaysDilbert">
    <s:metadata>
      <s:mimeTypes>
        <s:mimeType>image/*</s:mimeType>
      </s:mimeTypes>
    </s:metadata>
  </s:sink>
</s:scufl>
```

XScufle Schema

This document describes the elements and properties that can occur in a workflow definition file using the XScufl XML syntax. This syntax is used by the Taverna project to store and retrieve workflow definitions. Although this document exists, we do not recommend use of the XScufl format without the associated object model. The language itself should be considered volatile, any users of it should join the taverna developers list and monitor it, we do not support this language for use outside of Taverna.

This document applies to code from beta 10 onwards.

1.1. Scufl top level tag

Top level tag for an XScufl definition, contains arbitrary numbers of processor, link, source, sink and coordination elements defining member components of this workflow. This and all other tags described here are in the namespace http://org.embl.ebi.escience/xscufl/0.1alpha with the exception of the iteration strategy elements which are in their own namespace as detailed in that section.

- <scufl version(string) log(int)>
  - <processor>*
  - <link>*
  - <source>*
  - <sink>*
  - <coordination>*

1.2. Processor definition blocks

Defines a single atomic processing operation within a workflow. The extensibility element allows for definition of arbitrary plugin implementations, current (beta9) extensibility elements are listed below the main specification.

- <processor name(string)>
  - <description>?
    - textual description (PCDATA)
  - <SPEC ELEMENT maxretries(int)? retrydelay(int)? retrybackoff(double)?>
  - <iterationstrategy>?
  - <alternate>*
    - <SPEC ELEMENT maxretries(int)? retrydelay(int)? retrybackoff(double)?>
    - <outputmap key(string) value(string)>*
    - <inputmap key(string) value(string)>*

1.2.1.Processor retry properties

If the retry behaviour options are omited, they are assumed to default to values as shown below. These properties are specified seperately for the primary and any alternate processor implementations defined here.

i. maxretries = 0
ii. retrydelay = 0
iii. retrybackoff = 1.0

1.2.2.Alternate properties

If any alternates are specified, each must contain exactly one spec element and zero or more input and output mapping elements. These mapping elements are used to connect the ports of the top level processor with those of the alternate. The 'key' property contains the name of the appropriate input or output port on the original processor, the 'value' for each mapping contains the port name on the alternate that this input or output should be routed to or from.

1.2.3.Iteration strategy definition

This block, if present, defines how Taverna should behave in the event of type mismatches between expected and received input data objects. The default behaviour, if this block is not present, is to create an orthogonal join of all input data.

The iteration strategy works by creating a tree of iterators, with leaf nodes representing the iteration performed over a single input object, and non leaf nodes being combinations of their children according to some defined pattern. Currently there are two patterns, dot and cross products. In the case of the dot product, iteration over the dot node causes a simultaneous iteration over all child nodes, effectively moving through them in step and making the

implicit assumption that items in the child iterators have some inherent relationship. The cross product instead causes all possible combinations of the child iterators to be used.

Tags within the iterationstrategy element are in namespace :

http://org.embl.ebi.escience/xscufliteration/0.1beta10

- <iterationstrategy>
  - <dot> | <cross>
    - <dot>* | <cross>* | <iterator name(string)>*

The leaf 'iterator' nodes have a mandatory name attribute, this is the input port name that they will iterate over should this be required. In order to be valid, all bound ports for a given processor must have corresponding iterator elements in the iteration strategy. The first element inside the iterationstrategy element must be either a dot or a cross node, other than that arbitrary nesting of these combination nodes is permitted. This allows, for example, for an iteration strategy that combines two inputs together using a dot node and then generates all combinations of these pairs with a third input using a cross node.

More detailed examples of the iteration strategy will be in the beta10 release, this functionality is not present in beta9 release version, and was added to code in CVS on 5th April 2004.

1.3. Processor Spec elements

Available spec elements specified by the set of plugins provided with the beta9 release are shown below. Note that in beta9 the workflow plugin is non functional, and the biomoby one in a very early stage of development. These have both been improved on in the latest code in CVS, and should be fully operational in the beta10 release.

Available <SPEC ELEMENT> values in beta10	
org.embl.ebi.escience.scuflworkers.StringConstantProcessor	
• <stringconstant> o constant value (PCDATA)	Defines a string constant, commonly used as a paramete configuration option to avoid requesting commonly used v from the user each invocation. There are no inputs and a s output called 'value'
org.embl.ebi.escience.scuflworkers.wsdl.WSDLBasedProcessor	
• <arbitrarywsdl> o <wsdl> ▪ wsdl location (PCDATA) o <operation> ▪ operation name (PCDATA)	Defines access to a standard SOAP based web service refer by the URL to a WSDL document and the operation name w that document to access. Input and output ports are deter by introspection over the WSDL specification document.

org.embl.ebi.escience.scuflworkers.java.LocalServiceProcessor	
• <local> o class name (PCDATA)	Defines an operation running in the enactor's address based on a local Java class. This class implements org.embl.ebi.escience.scufworkers.java.LocalWorker inter Inputs and outputs are specified by the implementation cla
org.embl.ebi.escience.scuflworkers.soaplab.SoaplabProcessor	
• <soaplabwsdl> o application endpoint (PCDATA)	Defines an operation using a tool provided by Martin Sen Soaplab software. The application endpoint is the endpoi the specific application rather than the general Soaplab for example http://industry.ebi.ac.uk/soap/soaplab/alignment_multiple:
org.embl.ebi.escience.scuflworkers.biomoby.BiomobyProcessor	
• <biomobywsdl> o <mobyEndpoint> ▪ moby central endpoint (PCDATA) o <serviceName> ▪ service name (PCDATA) o <authorityName> ▪ authority name (PCDATA)	Defines an operation accessing a biomoby service. The central endpoint is used to locate the service with specified service and authority name. This endpoint is no endpoint of the service itself, it is the endpoint to the central service that can provide the details of the applic service specified. Input and output ports are derived service metadata. In the beta9 release this processor i fully implemented, beta10 will include full bi functionality.
org.embl.ebi.escience.scuflworkers.workflow.WorkflowProcessor	
• <workflow> o <xscufllocation>? ▪ definition location (PCDATA) o <xscufl>?	Defines an operation using a nested XScufl workflow. workflow is located using the URL specified in the defin location, inputs and outputs are generated to correspond t workflow source and sink elements defined in this defini As of beta10 it is possible to inline a workflow defin within a nested xscufl tag, the nested workflow follows same definition as any other and therefore allows arbi recursion. If a workflow is referenced and then modifi some editing operation the default behaviour is to copy modified workflow definition as an inline workflow, effect copy on write behaviour. Exactly one of xscufllocatio xscufl must be specified for this processor to be valid.
org.embl.ebi.escience.scuflworkers.talisman.TalismanProcessor	
• <talisman> o <tscript> ▪ script location (PCDATA)	Defines an operation based on the Talisman scripting en This is deprecated, and only of interest if you are alre heavy Talisman user, otherwise please disregard.

1.4. Data link declarations

Up to and including beta9 release, the syntax for connecting input and output ports on processors or overall workflow source / sink ports was more verbose than is described here, using nested elements rather than attributes.

1.4.1. Old syntax (beta9 and earlier)

- <link>
  - <input>
    - input / sink port name (PCDATA)
  - <output>
    - output / source port name (PCDATA)

1.4.2. New syntax (post beta9 release)

- <link source(string) sink(string)>

1.4.3. Port reference syntax

In both link syntax versions there are references to named input and output ports. Conceptually the link transfers data from an output port to an input port, regarding workflow sources as output ports (from the workflow perspective they are providing data) and workflow sinks as inputs (they receive data). There are two cases therefore that need to be defined, that of a port on a named processor, and that of a workflow source or sink.

1.4.3.1. Processor port syntax

- processorname:portname

1.4.3.2. Workflow source or sink syntax

- sourceorsinkname

For example, a link between output 'a' on processor 'foo' and input 'b' on processor 'bar would look like <link source="foo:a" sink="bar:b"/> and a link from the same output to a workflow sink called 'mysink' would be <link source="foo:a" sink="mysink"/>. The absence of a processor name qualifier is sufficient for Taverna to infer that the link is refering to a workflow sink or source depending on the context in which it occurs.

The change subsequent to beta9 of this syntax is largely to improve readability of the generated XML, and because people kept putting examples on posters. These examples were ugly, so various bits of code tidying ensued, of which this is one symptom.

1.5. Workflow source and sink declarations

These elements are used to declare top level workflow inputs and outputs, alternatively named sources and sinks respectively. They may optionally have a metadata block attached which describes the input or output; these are most relevent in the case of workflow outputs, as they are used to provide hints such as mime type information which can then be used to decorate the output data objects. In turn, these decorations are used to guide data visualisation operations further downstream.

- <sink> | <source>
  - source / sink name (PCDATA)
  - <metadata>?
    - <mimeTypes>?
      - <mimeType>*
        - mime type (PCDATA)
    - <description>?
      - textual description (PCDATA)
    - <semanticType>?
      - textual representation (PCDATA)

The metadata block may contain an arbitrary number of MIME type elements, if more than one exists then data objects are decorated with all types declared.

1.6. Coordination constraint declarations

Coordination constraints are used to prevent a processor transitioning between states until some constraint condition has been satisfied. While the specification here is generic, particular enactors may not be able to implement all constraint and gate combinations. Specifically, the myGrid enactor Freefluo is only capable of constraints that block a processor from transitioning from scheduled to running until another processor has completed. This is far and away the most common form desired in any case so this lack in the implementation causes few problems.

- <coordination name(string)>
  - <condition>
    - <state>
      - gate state (PCDATA)
    - <target>
      - controlling processor name (PCDATA)
  - <action>
    - <target>
      - controlled processor name (PCDATA)
    - <statechange>
      - <from>
        - initial state (PCDATA)
      - <to>
        - target state (PCDATA)

The states refered to here are a string enumeration, specifically values 'Completed', 'Scheduled' and 'Running' in the present implementation.

1.7. Examples

1.7.1.Example 1 - processor declaration and link elements

A workflow containing a single sequence fetcher processor. This processor makes use of the soaplab installation at the EBI and gets a single sequence from the 'seqret' EMBOSS tool. It contains a single workflow input and a single output with no additional typing information.

```
<?xml version="1.0" encoding="UTF-8"?>
<s:scufl xmlns:s="http://org.embl.ebi.escience/xscufl/0.1alpha" version="0.2"
log="0">
 <s:processor name="fetch">

<s:soaplabwsdl>http://industry.ebi.ac.uk/soap/soaplab/edit::seqret</s:soaplab
wsdl>
 </s:processor>
 <s:link source="sequenceID" sink="fetch:sequence_usa" />
 <s:link source="fetch:outseq" sink="sequenceString" />
 <s:source>sequenceID</s:source>
 <s:sink>sequenceString</s:sink>
</s:scufl>
```

1.7.2.Example 2 - processor declaration with alternate

A processor declaration including a single alternate instance. Both these processors (the primary and the alternate) use the local service mechanism, the primary is a test processor that always fails, the secondary is a simple operation that concatenates two strings together. As the port names for the primary and alternate processors are different, the inputmap and outputmap elements are required to map from primary port names (keys) to alternate port names (values).

```
<s:processor name="failingthing">
 <s:description>This processor always produces an error.</s:description>
 <s:local maxretries="2" retrydelay="1000" retrybackoff="2.0">
  org.embl.ebi.escience.scuflworkers.java.TestAlwaysFailingProcessor
 </s:local>
 <s:alternate>
  <s:local>
   org.embl.ebi.escience.scuflworkers.java.StringConcat
  </s:local>
  <s:outputmap key="urgle" value="output" />
  <s:inputmap key="foo" value="string1" />
  <s:inputmap key="bar" value="string2" />
 </s:alternate>
</s:processor>
```

1.7.3.Example 3 - coordination constraint declaration

A coordination constraint. This constraint states that the workflow engine should not allow the processor named 'destroyVizSession' to transition from state 'Scheduled' to state 'Running' until the processor named 'getDiagram'

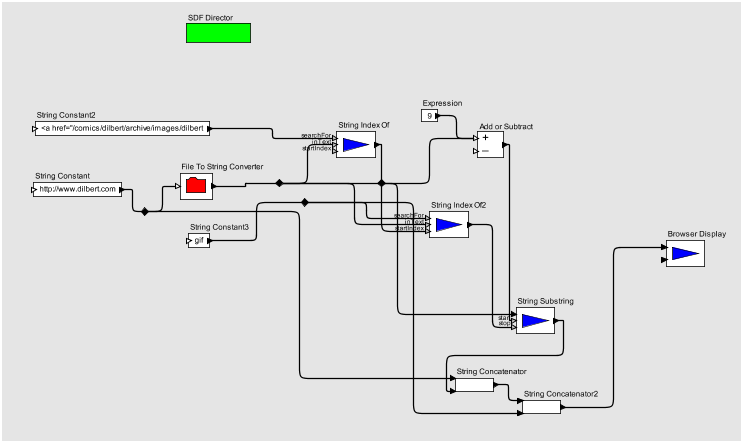
```
has attained state 'Completed'. Effectively this imposes an explicit temporal
ordering on the workflow invocation.

<s:coordination name="destroyVizSession_BLOCKON_getDiagram">
 <s:condition>
  <s:state>Completed</s:state>
  <s:target>getDiagram</s:target>
 </s:condition>
 <s:action>
  <s:target>destroyVizSession</s:target>
  <s:statechange>
   <s:from>Scheduled</s:from>
   <s:to>Running</s:to>
  </s:statechange>
 </s:action>
</s:coordination>
```

1.7.4.Example 4 - workflow output with metadata

```
A workflow sink or output, with associated metadata. This metadata block
states that data passing out of this workflow sink should be decorated with a
MIME type of 'text/x-graphviz', a free text description and a reference to a
term within the myGrid ontology. The syntax for a workflow source or input
with associated metadata would be identical, but with the 'source' tag in
place of the 'sink' tag here.

<s:sink>
 diagram
 <s:metadata>
  <s:mimeTypes>
   <s:mimeType>text/x-graphviz</s:mimeType>
  </s:mimeTypes>
  <s:description>
   A graphviz diagram showing the intersection of the
   workflow inputs by referencing against the gene ontology
  </s:description>
  <s:semanticType>
   http://www.mygrid.org.uk/ontology#diagram
  </s:semanticType>
 </s:metadata>
</s:sink>
```

MoML Analysis

Kepler Diagram



MoML Dilbert Workflow

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE entity PUBLIC "-//UC Berkeley//DTD MoML 1//EN"
    "http://ptolemy.eecs.berkeley.edu/xml/dtd/MoML_1.dtd">
<entity name="dilbertl" class="ptolemy.actor.TypedCompositeActor">
```

```
    <property name="_createdBy"
class="ptolemy.kernel.attributes.VersionAttribute" value="4.0.1">
    </property>
    <property name="SDF Director"
class="ptolemy.domains.sdf.kernel.SDFDirector">
        <property name="allowDisconnectedGraphs"
class="ptolemy.data.expr.Parameter" value="false">
        </property>
        <property name="allowRateChanges"
class="ptolemy.data.expr.Parameter" value="false">
        </property>
        <property name="iterations" class="ptolemy.data.expr.Parameter"
value="0">
        </property>
        <property name="vectorizationFactor"
class="ptolemy.data.expr.Parameter" value="1">
        </property>
        <property name="entityId"
class="org.ecoinformatics.seek.util.NamedObjId"
value="urn:lsid:ecoinformatics.org:kepler.300.1">
        </property>
        <property name="_location" class="ptolemy.kernel.util.Location"
value="[295.0, -65.0]">
        </property>
    </property>
    <property name="_windowProperties"
class="ptolemy.actor.gui.WindowPropertiesAttribute" value="{bounds={60,
135, 836, 785}, maximized=false}">
    </property>
    <property name="_vergilSize" class="ptolemy.actor.gui.SizeAttribute"
value="[506, 660]">
    </property>
    <property name="_vergilZoomFactor"
class="ptolemy.data.expr.ExpertParameter" value="1.0">
    </property>
    <property name="_vergilCenter"
class="ptolemy.data.expr.ExpertParameter" value="{903.0, 199.0}">
    </property>
    <entity name="String Constant" class="org.sdm.spa.StringConst">
        <property name="value" class="ptolemy.data.expr.FileParameter"
value="http://www.dilbert.com">
        </property>
        <property name="_icon" class="ptolemy.vergil.icon.BoxedValueIcon">
            <property name="attributeName"
class="ptolemy.kernel.util.StringAttribute" value="value">
            </property>
            <property name="displayWidth"
class="ptolemy.data.expr.Parameter" value="60">
            </property>
        </property>
        <property name="entityId"
class="org.ecoinformatics.seek.util.NamedObjId"
value="urn:lsid:ecoinformatics.org:kepler.204.1">
        </property>
        <property name="_location" class="ptolemy.kernel.util.Location"
value="{75.0, 180.0}">
        </property>
```

```
    </entity>
    <entity name="File To String Converter" class="org.geon.FileToString">
        <property name="fileOrURL" class="ptolemy.data.expr.FileParameter"
value="http://www.dilbert.com">
        </property>
        <property name="numberOfLinesToSkip"
class="ptolemy.data.expr.Parameter" value="0">
        </property>
        <property name="entityId"
class="org.ecoinformatics.seek.util.NamedObjId"
value="urn:lsid:ecoinformatics.org:kepler.143.1">
        </property>
        <property name="_location" class="ptolemy.kernel.util.Location"
value="[260.0, 175.0]">
        </property>
    </entity>
    <entity name="String Constant2" class="org.sdm.spa.StringConst">
        <property name="value" class="ptolemy.data.expr.FileParameter"
value="&lt;a href=&quot;/comics/dilbert/archive/images/dilbert">
        </property>
        <property name="_icon" class="ptolemy.vergil.icon.BoxedValueIcon">
            <property name="attributeName"
class="ptolemy.kernel.util.StringAttribute" value="value">
            </property>
            <property name="displayWidth"
class="ptolemy.data.expr.Parameter" value="60">
            </property>
        </property>
        <property name="entityId"
class="org.ecoinformatics.seek.util.NamedObjId"
value="urn:lsid:ecoinformatics.org:kepler.204.1">
        </property>
        <property name="_location" class="ptolemy.kernel.util.Location"
value="[145.0, 85.0]">
        </property>
    </entity>
    <entity name="String Index Of"
class="ptolemy.actor.lib.string.StringIndexOf">
        <property name="ignoreCase" class="ptolemy.data.expr.Parameter"
value="false">
        </property>
        <property name="startIndex"
class="ptolemy.actor.parameters.PortParameter" value="0">
        </property>
        <property name="searchForwards"
class="ptolemy.data.expr.Parameter" value="true">
        </property>
        <property name="entityId"
class="org.ecoinformatics.seek.util.NamedObjId"
value="urn:lsid:ecoinformatics.org:kepler.95.1">
        </property>
        <property name="_location" class="ptolemy.kernel.util.Location"
value="[510.0, 110.0]">
        </property>
    </entity>
    <entity name="String Index Of2"
class="ptolemy.actor.lib.string.StringIndexOf">
```

```
        <property name="ignoreCase" class="ptolemy.data.expr.Parameter"
value="false">
        </property>
        <property name="startIndex"
class="ptolemy.actor.parameters.PortParameter" value="0">
        </property>
        <property name="searchForwards"
class="ptolemy.data.expr.Parameter" value="true">
        </property>
        <property name="entityId"
class="org.ecoinformatics.seek.util.NamedObjId"
value="urn:lsid:ecoinformatics.org:kepler.95.1">
        </property>
        <property name="_location" class="ptolemy.kernel.util.Location"
value="[655.0, 235.0]">
        </property>
    </entity>
    <entity name="String Constant3" class="org.sdm.spa.StringConst">
        <property name="value" class="ptolemy.data.expr.FileParameter"
value="gif">
        </property>
        <property name="_icon" class="ptolemy.vergil.icon.BoxedValueIcon">
            <property name="attributeName"
class="ptolemy.kernel.util.StringAttribute" value="value">
            </property>
            <property name="displayWidth"
class="ptolemy.data.expr.Parameter" value="60">
            </property>
        </property>
        <property name="entityId"
class="org.ecoinformatics.seek.util.NamedObjId"
value="urn:lsid:ecoinformatics.org:kepler.204.1">
        </property>
        <property name="_location" class="ptolemy.kernel.util.Location"
value="{265.0, 260.0}">
        </property>
    </entity>
    <entity name="String Substring"
class="ptolemy.actor.lib.string.StringSubstring">
        <property name="start"
class="ptolemy.actor.parameters.PortParameter" value="0">
        </property>
        <property name="stop"
class="ptolemy.actor.parameters.PortParameter" value="0">
        </property>
        <property name="entityId"
class="org.ecoinformatics.seek.util.NamedObjId"
value="urn:lsid:ecoinformatics.org:kepler.99.1">
        </property>
        <property name="_location" class="ptolemy.kernel.util.Location"
value="[790.0, 385.0]">
        </property>
    </entity>
    <entity name="Add or Subtract" class="ptolemy.actor.lib.AddSubtract">
        <property name="entityId"
class="org.ecoinformatics.seek.util.NamedObjId"
value="urn:lsid:ecoinformatics.org:kepler.69.1">
```

```
        </property>
        <property name="_location" class="ptolemy.kernel.util.Location"
value="[720.0, 110.0]">
        </property>
    </entity>
    <entity name="Expression" class="ptolemy.actor.lib.Expression">
        <property name="expression"
class="ptolemy.kernel.util.StringAttribute" value="9">
        </property>
        <property name="entityId"
class="org.ecoinformatics.seek.util.NamedObjId"
value="urn:lsid:ecoinformatics.org:kepler.75.1">
        </property>
        <property name="_icon" class="ptolemy.vergil.icon.BoxedValueIcon">
            <property name="attributeName"
class="ptolemy.kernel.util.StringAttribute" value="expression">
            </property>
            <property name="displayWidth"
class="ptolemy.data.expr.Parameter" value="60">
            </property>
        </property>
        <property name="_location" class="ptolemy.kernel.util.Location"
value="{625.0, 65.0}">
        </property>
    </entity>
    <entity name="String Concatenator" class="org.sdm.spa.StringConcat">
        <property name="Pre" class="ptolemy.data.expr.Parameter"
value="&amp;quot;&amp;quot;">
        </property>
        <property name="Mid" class="ptolemy.data.expr.Parameter"
value="&amp;quot;&amp;quot;">
        </property>
        <property name="Post" class="ptolemy.data.expr.Parameter"
value="&amp;quot;&amp;quot;">
        </property>
        <property name="entityId"
class="org.ecoinformatics.seek.util.NamedObjId"
value="urn:lsid:ecoinformatics.org:kepler.203.1">
        </property>
        <property name="_location" class="ptolemy.kernel.util.Location"
value="[665.0, 475.0]">
        </property>
    </entity>
    <entity name="String Concatenator2" class="org.sdm.spa.StringConcat">
        <property name="Pre" class="ptolemy.data.expr.Parameter"
value="&amp;quot;&amp;quot;">
        </property>
        <property name="Mid" class="ptolemy.data.expr.Parameter"
value="&amp;quot;&amp;quot;">
        </property>
        <property name="Post" class="ptolemy.data.expr.Parameter"
value="&amp;quot;&amp;quot;">
        </property>
        <property name="entityId"
class="org.ecoinformatics.seek.util.NamedObjId"
value="urn:lsid:ecoinformatics.org:kepler.203.1">
        </property>
```

```
        <property name="_location" class="ptolemy.kernel.util.Location"
value="[770.0, 510.0]">
        </property>
    </entity>
    <entity name="Browser Display" class="org.geon.BrowserDisplay">
        <property name="entityId"
class="org.ecoinformatics.seek.util.NamedObjId"
value="urn:lsid:ecoinformatics.org:kepler.127.1">
        </property>
        <property name="_location" class="ptolemy.kernel.util.Location"
value="{1025, 280}">
        </property>
    </entity>
    <relation name="relation2" class="ptolemy.actor.TypedIORelation">
    </relation>
    <relation name="relation4" class="ptolemy.actor.TypedIORelation">
        <vertex name="vertex1" value="[390.0, 170.0]">
        </vertex>
    </relation>
    <relation name="relation7" class="ptolemy.actor.TypedIORelation">
    </relation>
    <relation name="relation8" class="ptolemy.actor.TypedIORelation">
        <vertex name="vertex1" value="[550.0, 170.0]">
        </vertex>
    </relation>
    <relation name="relation9" class="ptolemy.actor.TypedIORelation">
    </relation>
    <relation name="relation6" class="ptolemy.actor.TypedIORelation">
    </relation>
    <relation name="relation5" class="ptolemy.actor.TypedIORelation">
        <vertex name="vertex1" value="[180.0, 215.0]">
        </vertex>
    </relation>
    <relation name="relation11" class="ptolemy.actor.TypedIORelation">
    </relation>
    <relation name="relation10" class="ptolemy.actor.TypedIORelation">
        <vertex name="vertex1" value="{430.0, 200.0}">
        </vertex>
    </relation>
    <relation name="relation3" class="ptolemy.actor.TypedIORelation">
    </relation>
    <relation name="relation" class="ptolemy.actor.TypedIORelation">
    </relation>
    <link port="String Constant.output" relation="relation5"/>
    <link port="File To String Converter.output" relation="relation4"/>
    <link port="File To String Converter.trigger" relation="relation5"/>
    <link port="String Constant2.output" relation="relation2"/>
    <link port="String Index Of.searchFor" relation="relation2"/>
    <link port="String Index Of.inText" relation="relation4"/>
    <link port="String Index Of.output" relation="relation8"/>
    <link port="String Index Of2.searchFor" relation="relation10"/>
    <link port="String Index Of2.inText" relation="relation4"/>
    <link port="String Index Of2.startIndex" relation="relation8"/>
    <link port="String Index Of2.output" relation="relation7"/>
    <link port="String Constant3.output" relation="relation10"/>
    <link port="String Substring.input" relation="relation4"/>
    <link port="String Substring.output" relation="relation11"/>
```

```
        <link port="String Substring.start" relation="relation9"/>
        <link port="String Substring.stop" relation="relation7"/>
        <link port="Add or Subtract.plus" relation="relation8"/>
        <link port="Add or Subtract.plus" relation="relation6"/>
        <link port="Add or Subtract.output" relation="relation9"/>
        <link port="Expression.output" relation="relation6"/>
        <link port="String Concatenator.Result" relation="relation3"/>
        <link port="String Concatenator.String 1" relation="relation5"/>
        <link port="String Concatenator.String 2" relation="relation11"/>
        <link port="String Concatenator2.Result" relation="relation"/>
        <link port="String Concatenator2.String 1" relation="relation3"/>
        <link port="String Concatenator2.String 2" relation="relation10"/>
        <link port="Browser Display.inputURL" relation="relation"/>
    </entity>
```

MoML DTD and Schema

DTD

```
<!ELEMENT model (class | configure | director | doc | entity | import |
link | property | relation | rendition)*>
<!ATTLIST model name CDATA #REQUIRED
class CDATA #IMPLIED>
<!ELEMENT class (class | configure | director | doc | entity | import | link |
property | relation | rendition)*>
<!ATTLIST class name CDATA #REQUIRED
extends CDATA #IMPLIED>
<!ELEMENT configure (#PCDATA)>
<!ATTLIST configure source CDATA #IMPLIED>
<!ELEMENT director (configure | property)*>
<!ATTLIST director name CDATA "director"
class CDATA #REQUIRED>
<!ELEMENT doc (#PCDATA)>
<!ELEMENT entity (class | configure | director | doc | entity | import | link | port |
link | port | property | relation | rendition)*>
<!ATTLIST entity name CDATA #REQUIRED
class CDATA #IMPLIED>
<!ELEMENT import EMPTY>
<!ATTLIST import source CDATA #REQUIRED
base CDATA #IMPLIED>
<!ELEMENT link EMPTY>
<!ATTLIST link port CDATA #REQUIRED
relation CDATA #REQUIRED
vertex CDATA #IMPLIED>
<!ELEMENT location EMPTY>
<!ATTLIST location value CDATA #REQUIRED>
<!ELEMENT port (configure | doc | property)*>
<!ATTLIST port class CDATA #IMPLIED
name CDATA #REQUIRED>
<!ELEMENT property (configure | doc | property)*>
<!ATTLIST property class CDATA #IMPLIED
name CDATA #REQUIRED
value CDATA #IMPLIED>
<!ELEMENT relation (property | vertex)*>
<!ATTLIST relation name CDATA #REQUIRED
class CDATA #IMPLIED>
<!ELEMENT rendition (configure | location | property)*>
<!ATTLIST rendition class CDATA #REQUIRED>
<!ELEMENT vertex (location | property)*>
<!ATTLIST vertex name CDATA #REQUIRED
pathTo CDATA #IMPLIED>
```

Schema

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified">
  <xs:element name="model">
    <xs:complexType>
      <xs:choice minOccurs="0" maxOccurs="unbounded">
        <xs:element ref="class"/>
        <xs:element ref="configure"/>
        <xs:element ref="director"/>
        <xs:element ref="doc"/>
        <xs:element ref="entity"/>
        <xs:element ref="import"/>
        <xs:element ref="link"/>
        <xs:element ref="property"/>
        <xs:element ref="relation"/>
        <xs:element ref="rendition"/>
      </xs:choice>
      <xs:attributeGroup ref="attlist.model"/>
    </xs:complexType>
  </xs:element>
  <xs:attributeGroup name="attlist.model">
    <xs:attribute name="name" use="required"/>
    <xs:attribute name="class"/>
  </xs:attributeGroup>
  <xs:element name="class">
    <xs:complexType>
      <xs:choice minOccurs="0" maxOccurs="unbounded">
        <xs:element ref="class"/>
        <xs:element ref="configure"/>
        <xs:element ref="director"/>
        <xs:element ref="doc"/>
        <xs:element ref="entity"/>
        <xs:element ref="import"/>
        <xs:element ref="link"/>
        <xs:element ref="property"/>
        <xs:element ref="relation"/>
        <xs:element ref="rendition"/>
      </xs:choice>
      <xs:attributeGroup ref="attlist.class"/>
    </xs:complexType>
  </xs:element>
  <xs:attributeGroup name="attlist.class">
    <xs:attribute name="name" use="required"/>
    <xs:attribute name="extends"/>
  </xs:attributeGroup>
  <xs:element name="configure">
    <xs:complexType mixed="true">
      <xs:attributeGroup ref="attlist.configure"/>
    </xs:complexType>
  </xs:element>
  <xs:attributeGroup name="attlist.configure">
    <xs:attribute name="source"/>
  </xs:attributeGroup>
  <xs:element name="director">
    <xs:complexType>
      <xs:choice minOccurs="0" maxOccurs="unbounded">
        <xs:element ref="configure"/>
        <xs:element ref="property"/>
```

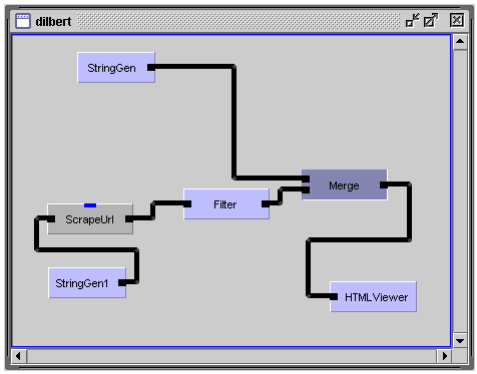
```
    </xs:choice>
    <xs:attributeGroup ref="attlist.director"/>
  </xs:complexType>
</xs:element>
<xs:attributeGroup name="attlist.director">
  <xs:attribute name="name" default="director"/>
  <xs:attribute name="class" use="required"/>
</xs:attributeGroup>
<xs:element name="doc" type="xs:string"/>
<xs:element name="entity">
  <xs:complexType>
    <xs:choice minOccurs="0" maxOccurs="unbounded">
      <xs:element ref="class"/>
      <xs:element ref="configure"/>
      <xs:element ref="director"/>
      <xs:element ref="doc"/>
      <xs:element ref="entity"/>
      <xs:element ref="import"/>
      <xs:element ref="link"/>
      <xs:element ref="port"/>
      <xs:element ref="link"/>
      <xs:element ref="port"/>
      <xs:element ref="property"/>
      <xs:element ref="relation"/>
      <xs:element ref="rendition"/>
    </xs:choice>
    <xs:attributeGroup ref="attlist.entity"/>
  </xs:complexType>
</xs:element>
<xs:attributeGroup name="attlist.entity">
  <xs:attribute name="name" use="required"/>
  <xs:attribute name="class"/>
</xs:attributeGroup>
<xs:element name="import">
  <xs:complexType>
    <xs:attributeGroup ref="attlist.import"/>
  </xs:complexType>
</xs:element>
<xs:attributeGroup name="attlist.import">
  <xs:attribute name="source" use="required"/>
  <xs:attribute name="base"/>
</xs:attributeGroup>
<xs:element name="link">
  <xs:complexType>
    <xs:attributeGroup ref="attlist.link"/>
  </xs:complexType>
</xs:element>
<xs:attributeGroup name="attlist.link">
  <xs:attribute name="port" use="required"/>
  <xs:attribute name="relation" use="required"/>
  <xs:attribute name="vertex"/>
</xs:attributeGroup>
<xs:element name="location">
  <xs:complexType>
    <xs:attributeGroup ref="attlist.location"/>
  </xs:complexType>
</xs:element>
```

```
<xs:attributeGroup name="attlist.location">
  <xs:attribute name="value" use="required"/>
</xs:attributeGroup>
<xs:element name="port">
  <xs:complexType>
    <xs:choice minOccurs="0" maxOccurs="unbounded">
      <xs:element ref="configure"/>
      <xs:element ref="doc"/>
      <xs:element ref="property"/>
    </xs:choice>
    <xs:attributeGroup ref="attlist.port"/>
  </xs:complexType>
</xs:element>
<xs:attributeGroup name="attlist.port">
  <xs:attribute name="class"/>
  <xs:attribute name="name" use="required"/>
</xs:attributeGroup>
<xs:element name="property">
  <xs:complexType>
    <xs:choice minOccurs="0" maxOccurs="unbounded">
      <xs:element ref="configure"/>
      <xs:element ref="doc"/>
      <xs:element ref="property"/>
    </xs:choice>
    <xs:attributeGroup ref="attlist.property"/>
  </xs:complexType>
</xs:element>
<xs:attributeGroup name="attlist.property">
  <xs:attribute name="class"/>
  <xs:attribute name="name" use="required"/>
  <xs:attribute name="value"/>
</xs:attributeGroup>
<xs:element name="relation">
  <xs:complexType>
    <xs:choice minOccurs="0" maxOccurs="unbounded">
      <xs:element ref="property"/>
      <xs:element ref="vertex"/>
    </xs:choice>
    <xs:attributeGroup ref="attlist.relation"/>
  </xs:complexType>
</xs:element>
<xs:attributeGroup name="attlist.relation">
  <xs:attribute name="name" use="required"/>
  <xs:attribute name="class"/>
</xs:attributeGroup>
<xs:element name="rendition">
  <xs:complexType>
    <xs:choice minOccurs="0" maxOccurs="unbounded">
      <xs:element ref="configure"/>
      <xs:element ref="location"/>
      <xs:element ref="property"/>
    </xs:choice>
    <xs:attributeGroup ref="attlist.rendition"/>
  </xs:complexType>
</xs:element>
<xs:attributeGroup name="attlist.rendition">
  <xs:attribute name="class" use="required"/>
```

```
  </xs:attributeGroup>
  <xs:element name="vertex">
    <xs:complexType>
      <xs:choice minOccurs="0" maxOccurs="unbounded">
        <xs:element ref="location"/>
        <xs:element ref="property"/>
      </xs:choice>
      <xs:attributeGroup ref="attlist.vertex"/>
    </xs:complexType>
  </xs:element>
  <xs:attributeGroup name="attlist.vertex">
    <xs:attribute name="name" use="required"/>
    <xs:attribute name="pathTo"/>
  </xs:attributeGroup>
</xs:schema>
```

SWFL Analysis

Triana Diagram



SWFL Dilbert Workflow

```
<?xml version="1.0" encoding="UTF-8"?>
<tool>
  <toolname>dilbert</toolname>
  <package />
  <inportnum>0</inportnum>
  <outportnum>0</outportnum>
  <inparam />
  <outparam />
  <parameters>
    <param name="popUpDescription" type="unknown">
      <value>No description for tool</value>
    </param>
  </parameters>
  <tasks>
    <task>
      <toolname>StringGen</toolname>
      <package>Common.String</package>
      <proxy type="Java">
        <param paramname="unitPackage">
          <value>Common.Input</value>
        </param>
        <param paramname="unitName">
          <value>Common.Input.StringGen</value>
        </param>
      </proxy>
      <inportnum>0</inportnum>
      <outportnum>1</outportnum>
      <inparam />
      <outparam />
      <output>
```

```
      <type>java.lang.String</type>
    </output>
    <parameters>
      <param name="maxOut" type="internal">
        <value>2147483647</value>
      </param>
      <param name="popUpDescription" type="internal">
        <value>Generates a String</value>
      </param>
      <param name="defaultIn" type="internal">
        <value>0</value>
      </param>
      <param name="toolVersion" type="internal">
        <value>3</value>
      </param>
      <param name="minIn" type="internal">
        <value>0</value>
      </param>
      <param name="outputType0" type="unknown">
        <value>java.lang.String</value>
      </param>
      <param name="defaultOut" type="internal">
        <value>1</value>
      </param>
      <param name="paramUpdatePolicy" type="internal">
        <value>processUpdate</value>
      </param>
      <param name="paramPanelInstantiate" type="internal">
        <value>onUserAccess</value>
      </param>
      <param name="maxIn" type="internal">
        <value>0</value>
      </param>
      <param name="minOut" type="internal">
        <value>0</value>
      </param>
      <param name="str" type="userAccessible">
        <value>http://www.dilbert.com</value>
      </param>
      <param name="paramPanelClass" type="internal">
        <value>StringGenPanel</value>
      </param>
      <param name="helpFile" type="internal">
        <value>StringGen.html</value>
      </param>
      <param name="guiY" type="gui">
        <value>0.21621621621621623</value>
      </param>
      <param name="guiX" type="gui">
        <value>0.4954954954954955</value>
      </param>
    </parameters>
  </task>
  <task>
    <toolname>Filter</toolname>
    <package>Common.String</package>
    <proxy type="Java">
```

```
        <param paramname="unitPackage">
          <value>Common.String</value>
        </param>
        <param paramname="unitName">
          <value>Common.String.Filter</value>
        </param>
      </proxy>
      <inportnum>1</inportnum>
      <outportnum>1</outportnum>
      <inparam />
      <outparam />
      <input>
        <type>java.lang.String</type>
      </input>
      <output>
        <type>java.lang.String</type>
      </output>
      <parameters>
        <param name="maxOut" type="internal">
          <value>2147483647</value>
        </param>
        <param name="popUpDescription" type="internal">
          <value>Only outputs strings that match a regular expression</value>
        </param>
        <param name="regex" type="userAccessible">
          <value>=(.*gif)</value>
        </param>
        <param name="defaultIn" type="internal">
          <value>1</value>
        </param>
        <param name="toolVersion" type="internal">
          <value>3</value>
        </param>
        <param name="minIn" type="internal">
          <value>1</value>
        </param>
        <param name="outputType0" type="unknown">
          <value>java.lang.String</value>
        </param>
        <param name="defaultOut" type="internal">
          <value>1</value>
        </param>
        <param name="paramUpdatePolicy" type="internal">
          <value>processUpdate</value>
        </param>
        <param name="guiBuilder" type="internal">
          <value>&lt;?jsx version="1"?&gt;&amp;nl&lt;java.lang.String
valueOf="Regular Expression $title regex TextField
&amp;nl"/&gt;&amp;nl&amp;nl</value>
        </param>
        <param name="maxIn" type="internal">
          <value>1</value>
        </param>
        <param name="minOut" type="internal">
          <value>0</value>
        </param>
        <param name="helpFile" type="internal">
```

```
      <value>Filter.html</value>
    </param>
    <param name="guiY" type="gui">
      <value>3.8378378378378377</value>
    </param>
    <param name="guiX" type="gui">
      <value>1.4234234234234233</value>
    </param>
  </parameters>
</task>
<task>
  <toolname>Merge</toolname>
  <package>Common.Stream</package>
  <proxy type="Java">
    <param paramname="unitPackage">
      <value>Common.Sync</value>
    </param>
    <param paramname="unitName">
      <value>Common.Sync.Merge</value>
    </param>
  </proxy>
  <inportnum>2</inportnum>
  <outportnum>1</outportnum>
  <inparam />
  <outparam />
  <input>
    <type>java.lang.Object</type>
  </input>
  <output>
    <type>java.lang.Object</type>
  </output>
  <parameters>
    <param name="maxOut" type="internal">
      <value>2147483647</value>
    </param>
    <param name="popUpDescription" type="internal">
      <value>Merges multiple streams of data</value>
    </param>
    <param name="defaultIn" type="internal">
      <value>2</value>
    </param>
    <param name="toolVersion" type="internal">
      <value>3</value>
    </param>
    <param name="minIn" type="internal">
      <value>0</value>
    </param>
    <param name="outputType0" type="unknown">
      <value>java.lang.String</value>
    </param>
    <param name="defaultOut" type="internal">
      <value>1</value>
    </param>
    <param name="paramUpdatePolicy" type="internal">
      <value>processUpdate</value>
    </param>
    <param name="defaultNodeRequirement" type="internal">
```

```
          <value>optional</value>
        </param>
        <param name="guiBuilder" type="internal">
          <value>&lt;?jsx version="1"?&gt;&amp;nl&lt;java.lang.String
valueOf="Merge Node Order (e.g. 1212) $title order TextField
&amp;nl"/&gt;&amp;nl&amp;nl</value>
        </param>
        <param name="maxIn" type="internal">
          <value>2147483647</value>
        </param>
        <param name="minOut" type="internal">
          <value>0</value>
        </param>
        <param name="order" type="userAccessible">
          <value />
        </param>
        <param name="helpFile" type="internal">
          <value>Merge.html</value>
        </param>
        <param name="guiY" type="gui">
          <value>3.324324324324324</value>
        </param>
        <param name="guiX" type="gui">
          <value>2.4594594594594597</value>
        </param>
      </parameters>
    </task>
    <task>
      <toolname>ScrapeUrl</toolname>
      <package>WebServices.Screen Scraper</package>
      <proxy type="WebService">
        <param paramname="portName">
          <value>ScreenScraperSoap</value>
        </param>
        <param paramname="wsdlLocation">
          <value>http://www.atomic-
x.com/xmlservices/screenscraper.asmx?wsdl#ScreenScraperSoap</value>
        </param>
        <param paramname="serializedPipe">
          <value>http://www.atomic-
x.com/xmlservices/screenscraper.asmx?wsdl#ScreenScraperSoap:ScreenScraperSoap
:ScrapeUrl</value>
        </param>
        <param paramname="operation">
          <value>ScrapeUrl</value>
        </param>
      </proxy>
      <renderingHints>
        <renderingHint hint="WebService" proxyDependent="true" />
      </renderingHints>
      <inportnum>1</inportnum>
      <outportnum>1</outportnum>
      <inparam />
      <outparam />
      <input>
        <type>Unknown Type</type>
      </input>
```

```
      <output>
        <type>Unknown Type</type>
      </output>
      <parameters>
        <param name="minOut" type="unknown">
          <value>1</value>
        </param>
        <param name="helpFile" type="unknown">
          <value>http://www.atomic-
x.com/xmlservices/screenscraper.asmx?wsdl#ScreenScraperSoap</value>
        </param>
        <param name="popUpDescription" type="unknown">
          <value>No description for tool</value>
        </param>
        <param name="maxIn" type="unknown">
          <value>1</value>
        </param>
        <param name="maxOut" type="unknown">
          <value>1</value>
        </param>
        <param name="guiY" type="gui">
          <value>4.216216216216216</value>
        </param>
        <param name="guiX" type="gui">
          <value>0.23423423423423423</value>
        </param>
        <param name="defaultOut" type="unknown">
          <value>1</value>
        </param>
        <param name="minIn" type="unknown">
          <value>1</value>
        </param>
        <param name="defaultIn" type="unknown">
          <value>1</value>
        </param>
      </parameters>
    </task>
    <task>
      <toolname>HTMLViewer</toolname>
      <package>Common.String</package>
      <proxy type="Java">
        <param paramname="unitPackage">
          <value>Common.String</value>
        </param>
        <param paramname="unitName">
          <value>Common.String.HTMLViewer</value>
        </param>
      </proxy>
      <inportnum>1</inportnum>
      <outportnum>0</outportnum>
      <inparam />
      <outparam />
      <input>
        <type>java.lang.Object</type>
      </input>
      <parameters>
        <param name="maxOut" type="internal">
```

```
      <value>0</value>
    </param>
    <param name="popUpDescription" type="internal">
      <value>View any HTML document or http address</value>
    </param>
    <param name="defaultIn" type="internal">
      <value>1</value>
    </param>
    <param name="toolVersion" type="internal">
      <value>3</value>
    </param>
    <param name="minIn" type="internal">
      <value>1</value>
    </param>
    <param name="defaultOut" type="internal">
      <value>0</value>
    </param>
    <param name="paramUpdatePolicy" type="internal">
      <value>processUpdate</value>
    </param>
    <param name="paramPanelInstantiate" type="internal">
      <value>onUserAccess</value>
    </param>
    <param name="defaultNodeRequirement" type="internal">
      <value>essentialIfConnected</value>
    </param>
    <param name="maxIn" type="internal">
      <value>1</value>
    </param>
    <param name="minOut" type="internal">
      <value>0</value>
    </param>
    <param name="str" type="userAccessible">
      <value />
    </param>
    <param name="paramPanelClass" type="internal">
      <value>HTMLViewPanel</value>
    </param>
    <param name="helpFile" type="internal">
      <value>HTMLViewer.html</value>
    </param>
    <param name="guiY" type="gui">
      <value>6.27027027027027</value>
    </param>
    <param name="guiX" type="gui">
      <value>2.7027027027027026</value>
    </param>
  </parameters>
</task>
<task>
  <toolname>StringGen1</toolname>
  <package>Common.String</package>
  <proxy type="Java">
    <param paramname="unitPackage">
      <value>Common.Input</value>
    </param>
    <param paramname="unitName">
```

```
    <value>Common.Input.StringGen</value>
  </param>
</proxy>
<inportnum>0</inportnum>
<outportnum>1</outportnum>
<inparam />
<outparam />
<output>
  <type>java.lang.String</type>
</output>
<parameters>
  <param name="maxOut" type="internal">
    <value>2147483647</value>
  </param>
  <param name="popUpDescription" type="internal">
    <value>Generates a String</value>
  </param>
  <param name="defaultIn" type="internal">
    <value>0</value>
  </param>
  <param name="toolVersion" type="internal">
    <value>3</value>
  </param>
  <param name="minIn" type="internal">
    <value>0</value>
  </param>
  <param name="outputType0" type="unknown">
    <value>java.lang.String</value>
  </param>
  <param name="defaultOut" type="internal">
    <value>1</value>
  </param>
  <param name="paramUpdatePolicy" type="internal">
    <value>processUpdate</value>
  </param>
  <param name="paramPanelInstantiate" type="internal">
    <value>onUserAccess</value>
  </param>
  <param name="maxIn" type="internal">
    <value>0</value>
  </param>
  <param name="minOut" type="internal">
    <value>0</value>
  </param>
  <param name="str" type="userAccessible">
    <value>http://www.dilbert.com</value>
  </param>
  <param name="paramPanelClass" type="internal">
    <value>StringGenPanel</value>
  </param>
  <param name="helpFile" type="internal">
    <value>StringGen.html</value>
  </param>
  <param name="guiY" type="gui">
    <value>5.918918918918919</value>
  </param>
  <param name="guiX" type="gui">
```

```
          <value>0.24324324324324326</value>
        </param>
      </parameters>
    </task>
    <connections>
      <connection>
        <source taskname="StringGen" node="0" />
        <target taskname="Merge" node="0" />
      </connection>
      <connection>
        <source taskname="ScrapeUrl" node="0" />
        <target taskname="Filter" node="0" />
      </connection>
      <connection>
        <source taskname="Filter" node="0" />
        <target taskname="Merge" node="1" />
      </connection>
      <connection>
        <source taskname="Merge" node="0" />
        <target taskname="HTMLViewer" node="0" />
      </connection>
      <connection>
        <source taskname="StringGen1" node="0" />
        <target taskname="ScrapeUrl" node="0" />
      </connection>
    </connections>
  </tasks>
</tool>
```

SWFL Schema

```
<?xml version="1.0" encoding="UTF-8"?>

<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
    xmlns:wsfl="http://www.cs.cf.ac.uk/User/Yan.Huang/xmlschema/wsfl/"
    xmlns:swfl="http://www.cs.cf.ac.uk/User/Yan.Huang/xmlschema/swfl/"
    targetNamespace="http://www.cs.cf.ac.uk/User/Yan.Huang/xmlschema/swfl/"
    elementFormDefault="qualified">

<xsd:element name="definitions" type="swfl:definitionsType">
    <xsd:unique name="swflFlowModelName">
        <xsd:selector xpath="swflFlowModel"/>
        <xsd:field xpath="@name"/>
    </xsd:unique>
</xsd:element>

<xsd:complexType name="definitionsType">
    <xsd:sequence>
        <xsd:element ref="swfl:swflFlowModel"
            minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
</xsd:complexType>
```

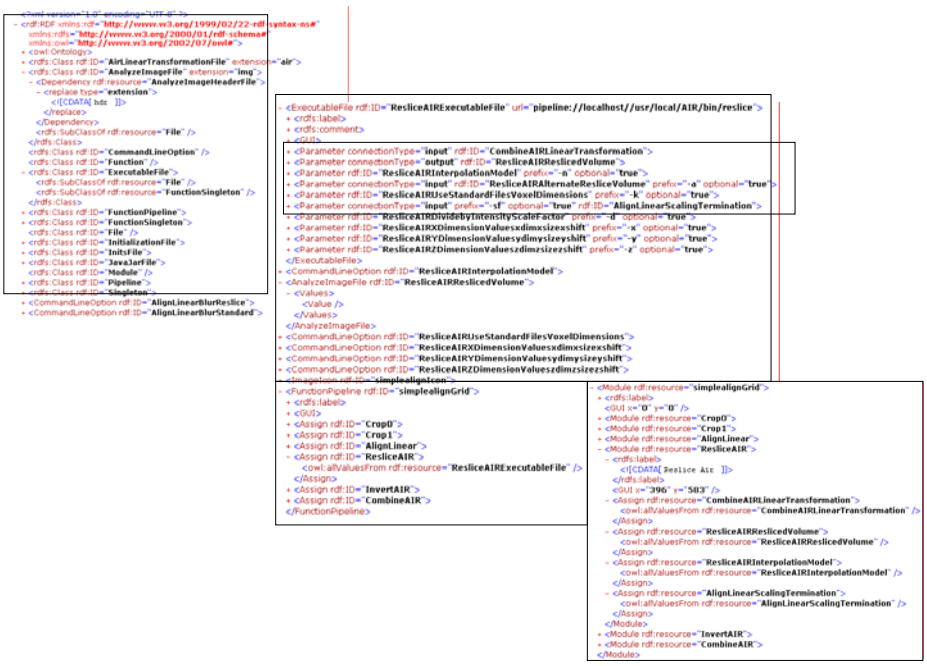
```
<xsd:element name="swflFlowModel" type="swfl:swflFlowModelType">
    <xsd:key name="providerName">
        <xsd:selector xpath="serviceProvider"/>
        <xsd:field xpath="@name"/>
    </xsd:key>
    <xsd:key name="activityName">
        <xsd:selector xpath="activity"/>
        <xsd:field xpath="@name"/>
    </xsd:key>
    <xsd:unique name="controlLinkName">
        <xsd:selector xpath="controlLink"/>
        <xsd:field xpath="@name"/>
    </xsd:unique>
    <xsd:unique name="dataLinkName">
        <xsd:selector xpath="dataLink"/>
        <xsd:field xpath="@name"/>
    </xsd:unique>
    <xsd:keyref name="linkActivityRef" refer="activityName">
        <xsd:selector xpath="implement|import"/>
        <xsd:field xpath="@source|@target"/>
    </xsd:keyref>
</xsd:element>

<xsd:complexType name="swflFlowModelType">
    <xsd:sequence>
        <xsd:element ref="wsdl:message"
            minOccurs="0"  maxOccurs="unbounded"/>
        <xsd:element name="flowSource" type="wsfl:flowSourceType"
            minOccurs="0"/>
        <xsd:element name="flowSink" type="wsfl:flowSinkType"
            minOccurs="0"/>
        <xsd:element name="serviceProvider" type="wsfl:serviceProviderType"
            minOccurs="0" maxOccurs="unbounded"/>
        <xsd:group ref="activityFlowGroup"/>
    </xsd:sequence>
    <xsd:attribute name="name" type="NCName" use="required"/>
    <xsd:attribute name="serviceProviderType" type="Qname"/>
</xsd:complexType>

<xsd:group name="activityFlowGroup">
    <xsd:sequence>
        <xsd:element name="activity" type="swflActivityType"
            minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element name="controlLink" type="swflControllinkType"
            minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element name="dataLink" type="swflDatalinkType"
            minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
</xsd:group>

<xsd:complexType name="swflActivityType">
    <xsd:choice>
        <xsd:element name="normal" type="wsfl:ActivityType"/>
        <xsd:element name="assign" type="assignmentType"/>
        <xsd:element name="if" type="controlType"/>
        <xsd:element name="while" type="loopType"/>
```

```
        <xsd:element name="dowhile" type="loopType"/>
        <xsd:element name="for" type="loopType"/>
        <xsd:element name="switch" type="controlType">
            <xsd:key name="CasePortName">
                <xsd:selector xpath="case"/>
                <xsd:field xpath="@port"/>
            </xsd:key>
        </xsd:element>
    </xsd:choice>
    <xsd:attribute name="operation" type="NCName"/>
</xsd:complexType>

<xsd:complexType name="assignmentType">
    <xsd:complexContent>
        <xsd:extension base="wsdl:tOperation">
            <xsd:sequence>
                <xsd:element name="left" type="dataPartType"/>
                <xsd:element name="right" type="dataPartType"/>
            </xsd:sequence>
            <xsd:attribute name="flowsource" type="NCName"/>
            <xsd:attribute name="part" type="NCName"/>
            <xsd:attribute name="converter" type="xsd:string"
use="optional"/>
            <xsd:attribute name="assignType" type="assignTypeType"/>
        </xsd:extension>
    </xsd:complexContent>
</xsd:complexType>

<xsd:simpleType name="assignTypeType">
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="="/>
        <xsd:enumeration value="+="/>
        <xsd:enumeration value="-="/>
        <xsd:enumeration value="*="/>
        <xsd:enumeration value="/="/>
    </xsd:restriction>
</xsd:simpleType>

<xsd:complexType name="loopType">
    <xsd:complexContent>
        <xsd:extension base="wsdl:tOperation">
            <xsd:sequence>
                <xsd:element name="expression" type="xsd:string"/>
            </xsd:sequence>
            <xsd:attribute name="setParallel" type="YesOrNoType"
                default="no" use="optional"/>
        </xsd:extension>
    </xsd:complexContent>
</xsd:complexType>


<xsd:complexType name="controlType">
    <xsd:complexContent>
        <xsd:extension base="wsdl:tOperation">
            <xsd:sequence>
                <xsd:element name="case" type="caseType"
                        minOccur="0" maxOccur="unbounded"/>
```

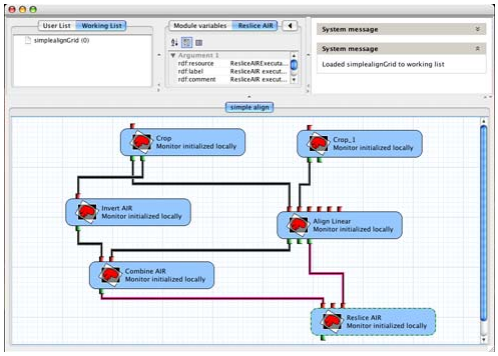
```
                <xsd:element name="defaultCase" type="defaultCaseType"
                        minOccur="0"/>
            </xsd:sequence>
            <xsd:attribute name="expression" type="String" use="optional"/>
        </xsd:extension>
    </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="caseType">
    <xsd:extension base="xsd:String">
        <xsd:attribute name="port" type="string" use="required"/>
    </xsd:extension>
</xsd:complexType>

<xsd:complexType name="defaultCaseType">
    <xsd:extension base="xsd:string">
        <xsd:attribute name="port" type="string" fixed="default"
use="required"/>
    </xsd:extension>
</xsd:complexType>

<xsd:complexType name="swflDatalinkType">
    <xsd:complexContent>
        <xsd:extension base="linkType">
            <xsd:sequence>
                <xsd:element ref="swflMap" minOccurs="0"/>
            </xsd:sequence>
        </xsd:extension>
    </xsd:complexContent>
</xsd:complexType>

<xsd:simpleType name="YesOrNoType">
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="yes"/>
        <xsd:enumeration value="no"/>
    </xsd:restriction>
</xsd:simpleType>

<xsd:complexType name="swflControllinkType">
    <xsd:complexContent>
            <xsd:extension base="wsfl:controlLinkType">
                <xsd:attribute name="controlPort" type="xsd:integer"
                        use="optional"/>
            </xsd:extension>
    </xsd:complexContent>
</xsd:complexType>

<xsd:element name="swflMap" type="swflMapType"/>
  <xsd:complexType name="swflMapType">
    <xsd:sequence>
        <xsd:element name="part" type="mapPartType"
                    minOccurs="1" maxOccurs="unbounded"/>
    </xsd:sequence>
    <xsd:attribute name="sourceMessage" type="NCName"/>
    <xsd:attribute name="targetMessage" type="NCName"/>
  </xsd:complexType>
```

```
  <xsd:complexType name="mapPartType">
    <sequence>
        <xsd:element name="sourcePart" type="dataPartType"/>
        <xsd:element name="targetpart" type="dataPartType"/>
    </sequence>
    <xsd:attribute name="name" type="NCName" use="optional"/>
    <xsd:attribute name="converter" type="NCName" use="optional"/>
    <xsd:attribute name="sharedType" type="YesOrNoType"
        default="no"  use="optional"/>
  </xsd:complexType>


  <xsd:complexType name="dataPartType">
    <xsd:sequence>
        <xsd:element name="item" type="itemType"
                    minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
    <xsd:attribute name="name" type="NCName" use="optional"/>
  </xsd:complexType>

  <xsd:complexType name="itemType">
    <xsd:choice>
        <xsd:element name="field">
            <xsd:complexType>
                <xsd:attribute name="name" type="xsd:string"/>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="index">
                    <xsd:complexType>
                <xsd:attribute name="dimension" type="xsd:integer"/>
                <xsd:attribute name="index" type="xsd:string"/>
            </xsd:complexType>
        </xsd:element>
    </xsd:choice>
  </xsd:complexType>

</xsd:schema>
```

LONI Analysis



LONI Diagram (not in standard representation)



LONI (Example, not Dilbert) Workflow

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
```

```
xmlns:owl="http://www.w3.org/2002/07/owl#">
 <owl:Ontology>
   <owl:versionInfo><![CDATA[20050412]]></owl:versionInfo>
 </owl:Ontology>
 <rdfs:Class rdf:ID="AirLinearTransformationFile" extension="air">
   <rdfs:SubClassOf rdf:resource="File" />
 </rdfs:Class>
 <rdfs:Class rdf:ID="AnalyzeImageFile" extension="img">
   <Dependency rdf:resource="AnalyzeImageHeaderFile">
     <replace type="extension"><![CDATA[hdr]]></replace>
   </Dependency>
   <rdfs:SubClassOf rdf:resource="File" />
 </rdfs:Class>
 <rdfs:Class rdf:ID="CommandLineOption" />
 <rdfs:Class rdf:ID="Function" />
 <rdfs:Class rdf:ID="ExecutableFile">
   <rdfs:SubClassOf rdf:resource="File" />
   <rdfs:SubClassOf rdf:resource="FunctionSingleton" />
 </rdfs:Class>
 <rdfs:Class rdf:ID="FunctionPipeline">
   <rdfs:SubClassOf rdf:resource="Function" />
 </rdfs:Class>
 <rdfs:Class rdf:ID="FunctionSingleton">
   <rdfs:SubClassOf rdf:resource="Function" />
 </rdfs:Class>
 <rdfs:Class rdf:ID="File" />
 <rdfs:Class rdf:ID="InitializationFile">
   <rdfs:SubClassOf rdf:resource="File" />
 </rdfs:Class>
 <rdfs:Class rdf:ID="InitsFile">
   <rdfs:SubClassOf rdf:resource="File" />
 </rdfs:Class>
 <rdfs:Class rdf:ID="JavaJarFile">
   <rdfs:SubClassOf rdf:resource="File" />
 </rdfs:Class>
 <rdfs:Class rdf:ID="Module" />
 <rdfs:Class rdf:ID="Pipeline">
   <rdfs:SubClassOf rdf:resource="Module" />
 </rdfs:Class>
 <rdfs:Class rdf:ID="Singleton">
   <rdfs:SubClassOf rdf:resource="Module" />
 </rdfs:Class>
 <CommandLineOption rdf:ID="AlignLinearBlurReslice">
   <Values>
     <Value>
       <string><![CDATA[0.1]]></string>
     </Value>
     <Value>
       <string><![CDATA[0.1]]></string>
     </Value>
     <Value>
       <string><![CDATA[0.1]]></string>
     </Value>
   </Values>
 </CommandLineOption>
 <CommandLineOption rdf:ID="AlignLinearBlurStandard">
   <Values>
```

```
      <Value>
        <string><![CDATA[0.1]]></string>
      </Value>
      <Value>
        <string><![CDATA[0.1]]></string>
      </Value>
      <Value>
        <string><![CDATA[0.1]]></string>
      </Value>
    </Values>
  </CommandLineOption>
  <CommandLineOption rdf:ID="AlignLinearConvergenceThreshold">
    <Values>
      <Value>
        <string><![CDATA[0.000010]]></string>
      </Value>
    </Values>
  </CommandLineOption>
  <CommandLineOption rdf:ID="AlignLinearCostFunction">
    <Values>
      <Value>
        <string><![CDATA[3]]></string>
      </Value>
    </Values>
  </CommandLineOption>
  <ExecutableFile rdf:ID="AlignLinearExecutableFile"
              url="pipeline://localhost//usr/local/AIR/bin/alignlinear">
      <rdfs:label><![CDATA[Align Linear]]></rdfs:label>
<rdfs:comment><![CDATA[This is a general linear intramodality registration
tool (within or across subjects, 2D or 3D). The user can specify any of a
variety of models, including rigid-body, affine, or perspective.

The program will generate a .air file that can be used to reslice the
specified reslice data set to match the specified standard data set.
]]></rdfs:comment>
      <GUI >
        <Icon rdf:resource="AIRIcon" />
        <rdfs:label><![CDATA[AlignLinear GUI]]></rdfs:label>
        <rdfs:comment><![CDATA[GUI for module AlignLinear]]></rdfs:comment>
      </GUI>
        <Parameter connectionType="input" rdf:ID="CropCroppedVolume">
          <rdfs:label><![CDATA[Standard Volume]]></rdfs:label>
          <rdfs:comment><![CDATA[This is the volume to which you are
aligning.]]></rdfs:comment>
        </Parameter>
        <Parameter connectionType="input" rdf:ID="Crop1CroppedVolume">
          <rdfs:label><![CDATA[Reslice Volume]]></rdfs:label>
          <rdfs:comment><![CDATA[This is the volume that is being
aligned.]]></rdfs:comment>
        </Parameter>
        <Parameter connectionType="output"
rdf:ID="AlignLinearLinearTransform">
          <rdfs:label><![CDATA[Linear Transform]]></rdfs:label>
          <rdfs:comment><![CDATA[A linear transformation matrix for aligning
the relice to the standard volume]]></rdfs:comment>
        </Parameter>
        <Parameter rdf:ID="AlignLinearModelNumber" prefix="-m">
```

```
          <rdfs:label><![CDATA[Model Number]]></rdfs:label>
          <rdfs:comment><![CDATA[Defines the degrees of freedom of the
transformation.
3-D models:
    6. rigid body 6 parameter model
    7. global rescale 7 parameter model
    9. traditional 9 parameter model (std must be on AC-PC line)
    12. affine 12 parameter model
    15. perspective 15 parameter model
2-D models (constrained to in-plane, no interpolation):
    23. 2-D rigid body 3 parameter model
    24. 2-D global rescale 4 parameter model
    25. 2-D affine/fixed determinant 5 parameter model
    26. 2-D affine 6 parameter model
    28. 2-D perspective 8 parameter model AIR 5.0
]]></rdfs:comment>
        </Parameter>
        <Parameter rdf:ID="AlignLinearThresholdStandard" prefix="-t1"
optional="true">
          <rdfs:label><![CDATA[Threshold Standard]]></rdfs:label>
          <rdfs:comment><![CDATA[The threshold for the standard image
defines a minimum voxel value for the standard file (-t1) or reslice file (-
t2). Voxels in the corresponding file with intensities below this value are
excluded from analysis when computing the cost function and its derivatives.
The value should always be an integer less than the maximum voxel value in
the corresponding file.
]]></rdfs:comment>
        </Parameter>
        <Parameter rdf:ID="AlignLinearThresholdReslice" prefix="-t2"
optional="true">
          <rdfs:label><![CDATA[Threshold Reslice]]></rdfs:label>
          <rdfs:comment><![CDATA[The threshold for the reslice image.

defines a minimum voxel value for the standard file (-t1) or reslice file (-
t2). Voxels in the corresponding file with intensities below this value are
excluded from analysis when computing the cost function and its derivatives.
The value should always be an integer less than the maximum voxel value in
the corresponding file.
]]></rdfs:comment>
        </Parameter>
        <Parameter rdf:ID="AlignLinearBlurStandard" prefix="-b1"
optional="true">
          <rdfs:label><![CDATA[Blur Standard]]></rdfs:label>
          <rdfs:comment><![CDATA[The Gaussian smoothing filter for the
standard image.

if this option is used, smoothing filters are applied along the x, y and z
axes of the standard file (-b1) or reslice file (-b2) before performing
registration. The FWHM value specifies the full width at half maximum of the
Gaussian smoothing filter to be applied along each dimension. The filters
have units of millimeters (or whatever units you use to specify voxel sizes
in your .hdr files). All three dimensions must be specified. If you give a
value of zero, no smoothing will be applied along the corresponding
dimension.
]]></rdfs:comment>
        </Parameter>
```

```
        <Parameter rdf:ID="AlignLinearBlurReslice" prefix="-b2"
optional="true">
          <rdfs:label><![CDATA[Blur Reslice]]></rdfs:label>
          <rdfs:comment><![CDATA[
The Gaussian smoothing filter for the reslice image.

if this option is used, smoothing filters are applied along the x, y and z
axes of the standard file (-b1) or reslice file (-b2) before performing
registration. The FWHM value specifies the full width at half maximum of the
Gaussian smoothing filter to be applied along each dimension. The filters
have units of millimeters (or whatever units you use to specify voxel sizes
in your .hdr files). All three dimensions must be specified. If you give a
value of zero, no smoothing will be applied along the corresponding
dimension.
]]></rdfs:comment>
        </Parameter>
        <Parameter rdf:ID="AlignLinearSegmentStandard" prefix="-p1"
optional="true">
          <rdfs:label><![CDATA[Segment Standard]]></rdfs:label>
          <rdfs:comment><![CDATA[Number of partitions for the standard image.

    defines the number of partitions used for segmenting the standard file in
the forward direction (-p1) or for segmenting the reslice file in the reverse
direction (-p2) when using the standard deviation of the ratio image as a
cost function. If this value is zero, no forward direction (-p1 0) or reverse
direction (-p2 0) computation is performed. A value of 256 is typically used
for intermodality registration when the corresponding file is an MRI study.
For MRI-PET registration, the number of partitions corresponding to the PET
study should be set to zero. When registering two dissimilar MR images, both
data sets can be assigned 256 partitions. For intramodality registration, the
default value of 1 is appropriate.
    only the ratio image cost function allows more than one partition
    AIR 5.0 Note that AIR now uses dynamic partitioning by default. Instead
of dividing the theoretical range of voxel intensities into the designated
number of partitions, the actual range of voxel intensities is divided into
this number of partitions. This assures that when the dynamic range of the
data is restricted, the voxels do not all end up occupying far fewer
partitions than requested. The -d flag forces the use of static partitioning
as in AIR 3.0x.
]]></rdfs:comment>
        </Parameter>
        <Parameter rdf:ID="AlignLinearSegmentReslice" prefix="-p2"
optional="true">
          <rdfs:label><![CDATA[Segment Reslice]]></rdfs:label>
          <rdfs:comment><![CDATA[
    defines the number of partitions used for segmenting the standard file in
the forward direction (-p1) or for segmenting the reslice file in the reverse
direction (-p2) when using the standard deviation of the ratio image as a
cost function. If this value is zero, no forward direction (-p1 0) or reverse
direction (-p2 0) computation is performed. A value of 256 is typically used
for intermodality registration when the corresponding file is an MRI study.
For MRI-PET registration, the number of partitions corresponding to the PET
study should be set to zero. When registering two dissimilar MR images, both
data sets can be assigned 256 partitions. For intramodality registration, the
default value of 1 is appropriate.
    only the ratio image cost function allows more than one partition
```

```
    AIR 5.0 Note that AIR now uses dynamic partitioning by default. Instead
of dividing the theoretical range of voxel intensities into the designated
number of partitions, the actual range of voxel intensities is divided into
this number of partitions. This assures that when the dynamic range of the
data is restricted, the voxels do not all end up occupying far fewer
partitions than requested. The -d flag forces the use of static partitioning
as in AIR 3.0x.
]]></rdfs:comment>
        </Parameter>
        <Parameter connectionType="input"
rdf:ID="AlignLinearStandardMaskVolume" prefix="-e1" optional="true">
          <rdfs:label><![CDATA[Standard Mask Volume]]></rdfs:label>
          <rdfs:comment><![CDATA[
this file is applied to the standard file (-e1) or reslice file (-e2) as a
mask. The file must match the corresponding file's dimensions, and voxels
that are zero in this file will be treated as if they were zero in the
corresponding file when computing the cost function. Mask files can be binary
or regular files.
]]></rdfs:comment>
        </Parameter>
        <Parameter connectionType="input"
rdf:ID="AlignLinearResliceMaskVolume" prefix="-e2" optional="true">
          <rdfs:label><![CDATA[Reslice Mask Volume]]></rdfs:label>
          <rdfs:comment><![CDATA[
this file is applied to the standard file (-e1) or reslice file (-e2) as a
mask. The file must match the corresponding file's dimensions, and voxels
that are zero in this file will be treated as if they were zero in the
corresponding file when computing the cost function. Mask files can be binary
or regular files.
]]></rdfs:comment>
        </Parameter>
        <Parameter connectionType="input" rdf:ID="AlignLinearInitialization"
prefix="-f" optional="true">
          <rdfs:label><![CDATA[Initialization file]]></rdfs:label>
          <rdfs:comment><![CDATA[
the name of an ASCII file containing spatial transformation initialization
parameters. These parameters can be used to control the starting position for
automated registration, a feature that is useful if the initial
misregistration is extreme (e.g., > 45Â° of rotational misregistration) or if
the default registration leads to an obviously incorrect result. The format
for the rigid-body initialization file is discussed under file types. Rigid-
body initialization files are created most easily using the program
manualreslice. Different spatial models require different numbers and types
of parameters in the initialization file. Note that some cost functions may
also allow an intensity parameter initialization file.
]]></rdfs:comment>
        </Parameter>
        <Parameter connectionType="output" rdf:ID="AlignLinearTermination"
prefix="-g" optional="true">
          <rdfs:label><![CDATA[Termination file]]></rdfs:label>
          <rdfs:comment><![CDATA[
the name of an ASCII file to be created containing spatial transformation
termination parameters. These parameters can then be used as initialization
parameters to restart the algorithm at the same location in parameter space
where it left off (using the same spatial model). This allows you to switch
cost functions or to vary smoothing, among other things. Different spatial
```

```
models create incompatible files. Note that some cost functions may also
allow an intensity parameter termination file.
]]></rdfs:comment>
        </Parameter>
        <Parameter connectionType="input"
rdf:ID="AlignLinearScalingInitialization" prefix="-fs" optional="true">
          <rdfs:label><![CDATA[Scaling Initialization file]]></rdfs:label>
          <rdfs:comment><![CDATA[
the name of an ASCII file containing a single parameter that initializes
intensity scaling
]]></rdfs:comment>
        </Parameter>
        <Parameter connectionType="output"
rdf:ID="AlignLinearScalingTermination" prefix="-gs" optional="true">
          <rdfs:label><![CDATA[Scaling Termination file]]></rdfs:label>
          <rdfs:comment><![CDATA[
the name of an ASCII file to be created containing the intensity scaling
parameter identified as optimal by the algorithm. Combined with a spatial
transformation initialization file, this parameter can be used to restart the
algorithm at the same location in parameter space where it left off. In
addition, the scaling parameter can be used as an intensity normalization
factor for subsequent statistical analysis of the registered data or as input
to reslice to create a final image that is intensity corrected as well as
spatially corrrected.
]]></rdfs:comment>
        </Parameter>
        <Parameter rdf:ID="AlignLinearSamplings" prefix="-s" optional="true">
          <rdfs:label><![CDATA[Samplings]]></rdfs:label>
          <rdfs:comment><![CDATA[
determines the number of intermediate iterative cycles of the algorithm. The
current sampling is divided by this ratio with each cycle to determine the
new sampling.
]]></rdfs:comment>
        </Parameter>
        <Parameter rdf:ID="AlignLinearConvergenceThreshold" prefix="-c"
optional="true">
          <rdfs:label><![CDATA[Convergence Threshold]]></rdfs:label>
          <rdfs:comment><![CDATA[
controls how small the predicted change in the cost function must be in order
to meet the convergence criteria. Setting this value too large will result in
convergence while the images are still misregistered; setting it too small
may lead to a failure to converge.

]]></rdfs:comment>
        </Parameter>
        <Parameter rdf:ID="AlignLinearRepeatedIterations" prefix="-r"
optional="true">
          <rdfs:label><![CDATA[Repeated Iterations]]></rdfs:label>
          <rdfs:comment><![CDATA[
controls the maximum number of iterations permitted at each sampling density.
If this number is made too low, it will lead to inaccurate results and/or
slow down the overall performance of the algorithm by preventing you from
making use of information that could have been derived more quickly at the
prematurely aborted, more superficial sampling.
]]></rdfs:comment>
        </Parameter>
```

```
        <Parameter rdf:ID="AlignLinearHaltAfterStepsWithoutImprovement"
prefix="-h" optional="true">
          <rdfs:label><![CDATA[Halt After Steps Without
Improvement]]></rdfs:label>
          <rdfs:comment><![CDATA[
controls the maximum number of iterations without any observed improvement in
the cost function. If greater than or equal to  repeated-iterations, this
value has no effect. At lower values, it can help you escape from situations
where you are bouncing back and forth between two or three locations in
parameter space without making any real progress. This sort of thing usually
only happens at superficial sampling densities.
]]></rdfs:comment>
        </Parameter>
        <Parameter rdf:ID="AlignLinearUseNonpositiveDefiniteHessianMatrices"
prefix="-j" optional="true">
          <rdfs:label><![CDATA[Use Non-positive Definite Hessian
Matrices]]></rdfs:label>
          <rdfs:comment><![CDATA[
uses currently unvalidated method for overcoming non-positive definite
Hessian matrices
]]></rdfs:comment>
        </Parameter>
        <Parameter rdf:ID="AlignLinearStaticPartitioningAIR30compatible"
prefix="-d" optional="true">
          <rdfs:label><![CDATA[Static Partitioning (AIR 3.0
compatible)]]></rdfs:label>
          <rdfs:comment><![CDATA[
forces use of static partitioning, reverting to default behavior of AIR 3.0x
]]></rdfs:comment>
        </Parameter>
        <Parameter
rdf:ID="AlignLinearNoninteractionofSpatialParameterDerivatives" prefix="-q"
optional="true">
          <rdfs:label><![CDATA[Non-interaction of Spatial Parameter
Derivatives]]></rdfs:label>
          <rdfs:comment><![CDATA[
assumes non-interaction of spatial parameter derivatives (reduces likelihood
of non-positive definite Hessian matrices)
]]></rdfs:comment>
        </Parameter>
        <Parameter rdf:ID="AlignLinearVerbose" prefix="-v" optional="true">
          <rdfs:label><![CDATA[Verbose option]]></rdfs:label>
          <rdfs:comment><![CDATA[
enables verbose reporting of interim results
]]></rdfs:comment>
        </Parameter>
        <Parameter rdf:ID="AlignLinearPrealignmentinterpolation" prefix="-z"
optional="true">
          <rdfs:label><![CDATA[Pre-alignment interpolation]]></rdfs:label>
          <rdfs:comment><![CDATA[
turns on pre-alignment interpolation to cubic voxels
]]></rdfs:comment>
        </Parameter>
        <Parameter rdf:ID="AlignLinearCostFunction" prefix="-x"
optional="true">
          <rdfs:label><![CDATA[Cost Function]]></rdfs:label>
          <rdfs:comment><![CDATA[
```

the number of the cost function selected from the following menu:

1. Standard deviation of ratio images cost function
    This cost function has the advantage of being independent of image intensity, so image intensities can be poorly matched and the registration will not be adversely affected. This is the only model that allows multiple partitions as required for intermodality registration.
2. Least squares cost function
    This cost function assumes that the image intensities are scaled identically. Least squares is computationally simpler and therefore faster than the standard deviation of ratio images, but may be inaccurate if the image intensities are poorly matched.
3. Least squares with global rescaling cost function
    This cost function is identical to the least squares cost function except that an intensity scaling term is added to the model.
]]></rdfs:comment>
        </Parameter>

  </ExecutableFile>

  <CommandLineOption rdf:ID="AlignLinearHaltAfterStepsWithoutImprovement">
    <Values>
      <Value>
        <string><![CDATA[5]]></string>
      </Value>
    </Values>
  </CommandLineOption>
  <ImageIcon rdf:ID="AIRIcon">
    <rdfs:label><![CDATA[AIR icon]]></rdfs:label>
    <rdfs:comment><![CDATA[AIR icon]]></rdfs:comment>
    <Values>
      <Value>
        <string><![CDATA[AIRlogo.gif]]></string>
      </Value>
    </Values>
  </ImageIcon>
  <InitializationFile rdf:ID="AlignLinearInitialization">
    <Values>
      <Value />
    </Values>
  </InitializationFile>
  <AirLinearTransformationFile rdf:ID="AlignLinearLinearTransform">
    <Values>
      <Value />
    </Values>
  </AirLinearTransformationFile>
  <CommandLineOption rdf:ID="AlignLinearModelNumber">
    <Values>
      <Value>
        <string><![CDATA[12]]></string>
      </Value>
    </Values>
  </CommandLineOption>

```
  <CommandLineOption
rdf:ID="AlignLinearNoninteractionofSpatialParameterDerivatives">
    <Values>
      <Value />
    </Values>
  </CommandLineOption>
  <CommandLineOption rdf:ID="AlignLinearPrealignmentinterpolation">
    <Values>
      <Value />
    </Values>
  </CommandLineOption>
  <CommandLineOption rdf:ID="AlignLinearRepeatedIterations">
    <Values>
      <Value>
        <string><![CDATA[25]]></string>
      </Value>
    </Values>
  </CommandLineOption>
  <AnalyzeImageFile rdf:ID="AlignLinearResliceMaskVolume">
    <Values>
      <Value />
    </Values>
  </AnalyzeImageFile>
  <CommandLineOption rdf:ID="AlignLinearSamplings">
    <Values>
      <Value>
        <string><![CDATA[81]]></string>
      </Value>
      <Value>
        <string><![CDATA[1]]></string>
      </Value>
      <Value>
        <string><![CDATA[3]]></string>
      </Value>
    </Values>
  </CommandLineOption>
  <InitializationFile rdf:ID="AlignLinearScalingInitialization">
    <Values>
      <Value />
    </Values>
  </InitializationFile>
  <InitsFile rdf:ID="AlignLinearScalingTermination">
    <Values>
      <Value />
    </Values>
  </InitsFile>
  <CommandLineOption rdf:ID="AlignLinearSegmentReslice">
    <Values>
      <Value>
        <string><![CDATA[1]]></string>
      </Value>
    </Values>
  </CommandLineOption>
  <CommandLineOption rdf:ID="AlignLinearSegmentStandard">
    <Values>
      <Value>
        <string><![CDATA[1]]></string>
```

```
      </Value>
    </Values>
  </CommandLineOption>
  <AnalyzeImageFile rdf:ID="AlignLinearStandardMaskVolume">
    <Values>
      <Value />
    </Values>
  </AnalyzeImageFile>
  <CommandLineOption rdf:ID="AlignLinearStaticPartitioningAIR30compatible">
    <Values>
      <Value />
    </Values>
  </CommandLineOption>
  <InitializationFile rdf:ID="AlignLinearTermination">
    <Values>
      <Value />
    </Values>
  </InitializationFile>
  <CommandLineOption rdf:ID="AlignLinearThresholdReslice">
    <Values>
      <Value>
        <string><![CDATA[1]]></string>
      </Value>
    </Values>
  </CommandLineOption>
  <CommandLineOption rdf:ID="AlignLinearThresholdStandard">
    <Values>
      <Value>
        <string><![CDATA[1]]></string>
      </Value>
    </Values>
  </CommandLineOption>
  <CommandLineOption
rdf:ID="AlignLinearUseNonpositiveDefiniteHessianMatrices">
    <Values>
      <Value />
    </Values>
  </CommandLineOption>
  <CommandLineOption rdf:ID="AlignLinearVerbose">
    <Values>
      <Value />
    </Values>
  </CommandLineOption>
  <CommandLineOption rdf:ID="CombineAIRCLIOption2">
    <Values>
      <Value>
        <string><![CDATA[y]]></string>
      </Value>
    </Values>
  </CommandLineOption>
  <ExecutableFile rdf:ID="CombineAIRExecutableFile"
              url="pipeline://localhost//usr/local/AIR/bin/combine_air">

      <rdfs:label><![CDATA[Combine AIR]]></rdfs:label>
<rdfs:comment><![CDATA[Given a .air file for registering file A to file B,
and another .air file for registering file B to file C, this program will
```

```
combine the two to create a .air file  that describes the direct
transformation of file A to file C.]]></rdfs:comment>
      <GUI>
        <Icon rdf:resource="AIRIcon" />
        <rdfs:label><![CDATA[CombineAIR GUI]]></rdfs:label>
        <rdfs:comment><![CDATA[GUI for module CombineAIR]]></rdfs:comment>
      </GUI>
        <Parameter connectionType="output"
rdf:ID="CombineAIRLinearTransformation">
          <rdfs:label><![CDATA[ Linear Transformation]]></rdfs:label>
          <rdfs:comment><![CDATA[name of the .air file to
create]]></rdfs:comment>
        </Parameter>
        <Parameter rdf:ID="CombineAIRCLIOption2">
          <rdfs:label><![CDATA[overwrite?(y/n) ]]></rdfs:label>
          <rdfs:comment><![CDATA['y' grants permission to overwrite
output]]></rdfs:comment>
        </Parameter>
        <Parameter connectionType="input"
rdf:ID="InvertAIRInvertedLinearTransformation">
          <rdfs:label><![CDATA[ Linear Transformation]]></rdfs:label>
          <rdfs:comment><![CDATA[name of the .air file that targets the final
space into which the images should be transformed]]></rdfs:comment>
        </Parameter>
        <Parameter connectionType="input"
rdf:ID="AlignLinearLinearTransform">
          <rdfs:label><![CDATA[ Linear Transformation]]></rdfs:label>
          <rdfs:comment><![CDATA[a .air file that has a standard file that is
spatially equivalent to the reslice file in air-file1Linear Transformation
for CombineAIR]]></rdfs:comment>
        </Parameter>

  </ExecutableFile>
  <AirLinearTransformationFile rdf:ID="CombineAIRLinearTransformation">
    <Values>
      <Value />
    </Values>
  </AirLinearTransformationFile>
  <CommandLineOption rdf:ID="Crop1CLIOption4">
    <Values>
      <Value>
        <string><![CDATA[y]]></string>
      </Value>
    </Values>
  </CommandLineOption>
  <AnalyzeImageFile rdf:ID="Crop1CroppedVolume">
    <Values>
      <Value />
    </Values>
  </AnalyzeImageFile>

  <CommandLineOption rdf:ID="Crop1Padding">
    <Values>
      <Value>
        <string><![CDATA[3]]></string>
      </Value>
    </Values>
```

```
  </CommandLineOption>
  <AirLinearTransformationFile rdf:ID="Crop1Transformation">
    <Values>
      <Value />
    </Values>
  </AirLinearTransformationFile>
  <AnalyzeImageFile rdf:ID="Crop1Volume">
    <Values>
      <Value bindValueFromURL="true">

<string><![CDATA[pipeline://localhost//loni/data/simpleAlignCropVolume.list]]
></string>
      </Value>
    </Values>
  </AnalyzeImageFile>
  <CommandLineOption rdf:ID="CropCLIOption4">
    <Values>
      <Value>
        <string><![CDATA[y]]></string>
      </Value>
    </Values>
  </CommandLineOption>
  <AnalyzeImageFile rdf:ID="CropCroppedVolume">
    <Values>
      <Value />
    </Values>
  </AnalyzeImageFile>
  <ExecutableFile rdf:ID="CropExecutableFile"
              url="pipeline://localhost//usr/local/AIR/bin/crop">
      <GUI>
        <Icon rdf:resource="AIRIcon" />
        <rdfs:label><![CDATA[Crop GUI]]></rdfs:label>
        <rdfs:comment><![CDATA[GUI for module Crop]]></rdfs:comment>
      </GUI>
        <Parameter connectionType="input" rdf:ID="CropVolume">
          <rdfs:label><![CDATA[Crop Volume]]></rdfs:label>
          <rdfs:comment><![CDATA[the name of the file to be
cropped]]></rdfs:comment>
        </Parameter>
        <Parameter rdf:ID="CropPadding">
          <rdfs:label><![CDATA[Crop Padding]]></rdfs:label>
          <rdfs:comment><![CDATA[the number of voxels of padding to add back
after cropping (more won't be added than was cropped)]]></rdfs:comment>
        </Parameter>
        <Parameter connectionType="output" rdf:ID="CropCroppedVolume">
          <rdfs:label><![CDATA[Crop Cropped Volume]]></rdfs:label>
          <rdfs:comment><![CDATA[the name of the output
file]]></rdfs:comment>
        </Parameter>
        <Parameter rdf:ID="CropCLIOption4">
          <rdfs:label><![CDATA[Crop overwrite?(y/n)]]></rdfs:label>
          <rdfs:comment><![CDATA['y' grants permission to overwrite
output]]></rdfs:comment>
        </Parameter>
        <Parameter connectionType="output" rdf:ID="CropTransformation"
optional="true">
          <rdfs:label><![CDATA[Crop Transformation]]></rdfs:label>
```

```
          <rdfs:comment><![CDATA[the name of a .air file that will be created
to describe the crop with input as the reslice file and output as the
standard file.]]></rdfs:comment>
        </Parameter>
    <rdfs:label><![CDATA[Crop executable]]></rdfs:label>
    <rdfs:comment><![CDATA[executable for Crop]]></rdfs:comment>
  </ExecutableFile>

  <CommandLineOption rdf:ID="CropPadding">
    <Values>
      <Value>
        <string><![CDATA[3]]></string>
      </Value>
    </Values>
  </CommandLineOption>
  <AirLinearTransformationFile rdf:ID="CropTransformation">
    <Values>
      <Value />
    </Values>
  </AirLinearTransformationFile>
  <AnalyzeImageFile rdf:ID="CropVolume">
    <Values>
      <Value>

<string><![CDATA[pipeline://localhost//loni/mbirn/EAE/Atlas/EAE_warp_atlas.im
g]]></string>
      </Value>
    </Values>
  </AnalyzeImageFile>
  <ExecutableFile rdf:ID="InvertAIRExecutableFile"
              url="pipeline://localhost//usr/local/AIR/bin/invert_air">
      <rdfs:label><![CDATA[Invert AIR]]></rdfs:label>
<rdfs:comment><![CDATA[This program will take a .air file for reslicing file
A to match file B and convert it to a .air file for reslicing file B to match
file A.]]></rdfs:comment>
      <GUI>
        <Icon rdf:resource="AIRIcon" />
        <rdfs:label><![CDATA[InvertAIR GUI]]></rdfs:label>
        <rdfs:comment><![CDATA[GUI for module InvertAIR]]></rdfs:comment>
      </GUI>
        <Parameter connectionType="input" rdf:ID="CropTransformation">
          <rdfs:label><![CDATA[Linear Transformation]]></rdfs:label>
          <rdfs:comment><![CDATA[the name of the .air file to be
inverted]]></rdfs:comment>
        </Parameter>
        <Parameter connectionType="output"
rdf:ID="InvertAIRInvertedLinearTransformation">
          <rdfs:label><![CDATA[Inverted Linear Transformation]]></rdfs:label>
          <rdfs:comment><![CDATA[the name of the new .air file to be
created]]></rdfs:comment>
        </Parameter>

  </ExecutableFile>

  <AirLinearTransformationFile
rdf:ID="InvertAIRInvertedLinearTransformation">
    <Values>
```

```
      <Value />
    </Values>
  </AirLinearTransformationFile>
  <AnalyzeImageFile rdf:ID="ResliceAIRAlternateResliceVolume">
    <Values>
      <Value />
    </Values>
  </AnalyzeImageFile>
  <CommandLineOption rdf:ID="ResliceAIRDividebyIntensityScaleFactor">
    <Values>
      <Value />
    </Values>
  </CommandLineOption>
  <ExecutableFile rdf:ID="ResliceAIRExecutableFile"
              url="pipeline://localhost//usr/local/AIR/bin/reslice">

      <rdfs:label><![CDATA[Reslice AIR]]></rdfs:label>
      <rdfs:comment><![CDATA[This program takes a .air file and uses the
information that it contains to load the corresponding image file and
generate a new, realigned file.]]></rdfs:comment>
      <GUI>
        <Icon rdf:resource="AIRIcon" />
        <rdfs:label><![CDATA[ResliceAIR GUI]]></rdfs:label>
        <rdfs:comment><![CDATA[GUI for module ResliceAIR]]></rdfs:comment>
      </GUI>

        <Parameter connectionType="input"
rdf:ID="CombineAIRLinearTransformation">
          <rdfs:label><![CDATA[Linear Transformation]]></rdfs:label>
          <rdfs:comment><![CDATA[name of the input .air file containing the
reslice parameters]]></rdfs:comment>
        </Parameter>
        <Parameter connectionType="output" rdf:ID="ResliceAIRReslicedVolume">
          <rdfs:label><![CDATA[Resliced Volume]]></rdfs:label>
          <rdfs:comment><![CDATA[name of the output image file (.hdr or .img
suffix optional)]]></rdfs:comment>
        </Parameter>
        <Parameter rdf:ID="ResliceAIRInterpolationModel" prefix="-n"
optional="true">
          <rdfs:label><![CDATA[Interpolation Model]]></rdfs:label>
          <rdfs:comment><![CDATA[
the interpolation model number, selected from the menu:

    * 0. nearest neighbor
    * 1. trilinear
    * 2. windowed sinc in original xy plane, linear along z
         o requires x- and y- half-window-width arguments
    * 3. windowed sinc in original xz plane. linear along y
         o requires x- and z- half-window-width arguments
    * 4. windowed sinc in original yz plane, linear along x
         o requires y- and z- half-window-width arguments
    * 5. 3D windowed sinc
         o requires x-, y- and z- half-window-width arguments
    * 6. 3D windowed scanline sinc
         o requires x-, y- and z- half-window-width arguments
    * 7. 3D unwindowed scanline sinc
    * 10. 3D scanline chirp-z
```

```
    * 11. scanline chirp-z in original xy plane, linear along z
    * 12. scanline chirp-z in original xz plane, linear along y
    * 13. scanline chirp-z in original yz plane, linear along
x]]></rdfs:comment>
        </Parameter>
        <Parameter connectionType="input"
rdf:ID="ResliceAIRAlternateResliceVolume" prefix="-a" optional="true">
          <rdfs:label><![CDATA[Alternate Reslice Volume]]></rdfs:label>
          <rdfs:comment><![CDATA[
name of an alternate reslice file that is spatially equivalent (same voxel
sizes and dimensions) to the default reslice file specified in the .air-
file]]></rdfs:comment>
        </Parameter>
        <Parameter rdf:ID="ResliceAIRUseStandardFilesVoxelDimensions"
prefix="-k" optional="true">
          <rdfs:label><![CDATA[Use Standard File's Voxel
Dimensions]]></rdfs:label>
          <rdfs:comment><![CDATA[
retains the standard file's voxel dimensions (default is to interpolate to
cubic voxels)]]></rdfs:comment>
        </Parameter>
        <Parameter connectionType="input" prefix="-sf" optional="true"
rdf:ID="AlignLinearScalingTermination">
          <rdfs:label><![CDATA[Multiplicative Intensity Scale Factor
File]]></rdfs:label>
          <rdfs:comment><![CDATA[output intensities are multiplied by the
value in this ASCII file containing a single positive
number]]></rdfs:comment>
        </Parameter>
        <Parameter rdf:ID="ResliceAIRDividebyIntensityScaleFactor" prefix="-
d" optional="true">
          <rdfs:label><![CDATA[Divide by Intensity Scale
Factor]]></rdfs:label>
          <rdfs:comment><![CDATA[output intensities are divided by this
positive value]]></rdfs:comment>
        </Parameter>
        <Parameter rdf:ID="ResliceAIRXDimensionValuesxdimxsizexshift"
prefix="-x" optional="true">
          <rdfs:label><![CDATA[X Dimension Values]]></rdfs:label>
          <rdfs:comment><![CDATA[
alters output x-dimension, x-voxel size or causes shift of output along the
x-axis  (x_dim x_size [x_shift])
]]></rdfs:comment>
        </Parameter>
        <Parameter rdf:ID="ResliceAIRYDimensionValuesydimysizeyshift"
prefix="-y" optional="true">
          <rdfs:label><![CDATA[Y Dimension Values (y_dim y_size
[y_shift])]]></rdfs:label>
          <rdfs:comment><![CDATA[alters output y-dimension, y-voxel size or
causes shift of output along the y-axis (y_dim y_size
[y_shift]]]></rdfs:comment>
        </Parameter>
        <Parameter rdf:ID="ResliceAIRZDimensionValueszdimzsizezshift"
prefix="-z" optional="true">
          <rdfs:label><![CDATA[Z Dimension Values]]></rdfs:label>
```

```
          <rdfs:comment><![CDATA[alters output z-dimension, z-voxel size or
causes shift of output along the z-axis  (z_dim z_size
[z_shift])]]></rdfs:comment>
        </Parameter>


  </ExecutableFile>

  <CommandLineOption rdf:ID="ResliceAIRInterpolationModel">
    <Values>
      <Value>
        <string><![CDATA[11]]></string>
      </Value>
    </Values>
  </CommandLineOption>
  <AnalyzeImageFile rdf:ID="ResliceAIRReslicedVolume">
    <Values>
      <Value />
    </Values>
  </AnalyzeImageFile>
  <CommandLineOption rdf:ID="ResliceAIRUseStandardFilesVoxelDimensions">
    <Values>
      <Value />
    </Values>
  </CommandLineOption>
  <CommandLineOption rdf:ID="ResliceAIRXDimensionValuesxdimxsizexshift">
    <Values>
      <Value />
    </Values>
  </CommandLineOption>
  <CommandLineOption rdf:ID="ResliceAIRYDimensionValuesydimysizeyshift">
    <Values>
      <Value />
    </Values>
  </CommandLineOption>
  <CommandLineOption rdf:ID="ResliceAIRZDimensionValueszdimzsizezshift">
    <Values>
      <Value />
    </Values>
  </CommandLineOption>
  <ImageIcon rdf:ID="simplealignIcon">
    <rdfs:label><![CDATA[simplealign icon]]></rdfs:label>
    <rdfs:comment><![CDATA[icon for module simplealign]]></rdfs:comment>
    <Values>
      <Value>
        <string />
      </Value>
    </Values>
  </ImageIcon>

  <FunctionPipeline rdf:ID="simplealignGrid">
    <rdfs:label><![CDATA[simple align]]></rdfs:label>
    <GUI>
      <Icon rdf:resource="simplealignIcon" />
    </GUI>
    <Assign rdf:ID="Crop0">
      <owl:allValuesFrom rdf:resource="CropExecutableFile" />
```

```
    </Assign>
    <Assign rdf:ID="Crop1">
      <owl:allValuesFrom rdf:resource="CropExecutableFile" />
    </Assign>
    <Assign rdf:ID="AlignLinear">
      <owl:allValuesFrom rdf:resource="AlignLinearExecutableFile" />
    </Assign>
    <Assign rdf:ID="ResliceAIR">
      <owl:allValuesFrom rdf:resource="ResliceAIRExecutableFile" />
    </Assign>
    <Assign rdf:ID="InvertAIR">
      <owl:allValuesFrom rdf:resource="InvertAIRExecutableFile" />
    </Assign>
    <Assign rdf:ID="CombineAIR">
      <owl:allValuesFrom rdf:resource="CombineAIRExecutableFile" />
    </Assign>
  </FunctionPipeline>


  <Module rdf:resource="simplealignGrid">
    <rdfs:label><![CDATA[simple align]]></rdfs:label>
    <GUI x="0" y="0">
    </GUI>
    <Module rdf:resource="Crop0">
    <rdfs:label><![CDATA[Crop0]]></rdfs:label>
      <rdfs:comment>Find the transform resulting from cropping the
standard</rdfs:comment>
      <GUI x="233" y="115">
      </GUI>
        <Assign rdf:resource="CropVolume">
        <owl:allValuesFrom rdf:resource="CropVolume" />
        </Assign>
        <Assign rdf:resource="CropPadding">
        <owl:allValuesFrom rdf:resource="CropPadding" />
        </Assign>
        <Assign rdf:resource="CropCroppedVolume">
        <owl:allValuesFrom rdf:resource="CropCroppedVolume" />
        </Assign>
        <Assign rdf:resource="CropCLIOption4">
        <owl:allValuesFrom rdf:resource="CropCLIOption4" />
        </Assign>
        <Assign rdf:resource="CropTransformation">
        <owl:allValuesFrom rdf:resource="CropTransformation" />
        </Assign>
    </Module>
    <Module rdf:resource="Crop1">
    <rdfs:label><![CDATA[Crop1]]></rdfs:label>
      <rdfs:comment>Crop a set of images</rdfs:comment>
      <GUI x="572" y="113">
      </GUI>
        <Assign rdf:resource="CropVolume">
        <owl:allValuesFrom rdf:resource="Crop1Volume" />
        </Assign>
        <Assign rdf:resource="CropPadding">
        <owl:allValuesFrom rdf:resource="Crop1Padding" />
        </Assign>
        <Assign rdf:resource="CropCroppedVolume">
```

```
        <owl:allValuesFrom rdf:resource="Crop1CroppedVolume" />
        </Assign>
        <Assign rdf:resource="CropCLIOption4">
        <owl:allValuesFrom rdf:resource="Crop1CLIOption4" />
        </Assign>
    </Module>
    <Module rdf:resource="AlignLinear">
    <rdfs:label><![CDATA[Align Linear]]></rdfs:label>
      <GUI x="418" y="308">
      </GUI>
        <Assign  rdf:resource="CropCroppedVolume">
        <owl:allValuesFrom rdf:resource="CropCroppedVolume" />
        </Assign>
        <Assign rdf:resource="Crop1CroppedVolume">
        <owl:allValuesFrom rdf:resource="Crop1CroppedVolume" />
        </Assign>
        <Assign rdf:resource="AlignLinearLinearTransform">
        <owl:allValuesFrom rdf:resource="AlignLinearLinearTransform" />
        </Assign>
        <Assign rdf:resource="AlignLinearModelNumber">
        <owl:allValuesFrom rdf:resource="AlignLinearModelNumber" />
        </Assign>
        <Assign rdf:resource="AlignLinearThresholdStandard">
        <owl:allValuesFrom rdf:resource="AlignLinearThresholdStandard" />
        </Assign>
        <Assign rdf:resource="AlignLinearThresholdReslice">
        <owl:allValuesFrom rdf:resource="AlignLinearThresholdReslice" />
        </Assign>
        <Assign rdf:resource="AlignLinearBlurStandard">
        <owl:allValuesFrom rdf:resource="AlignLinearBlurStandard" />
        </Assign>
        <Assign rdf:resource="AlignLinearBlurReslice">
        <owl:allValuesFrom rdf:resource="AlignLinearBlurReslice" />
        </Assign>
        <Assign rdf:resource="AlignLinearSegmentStandard">
        <owl:allValuesFrom rdf:resource="AlignLinearSegmentStandard" />
        </Assign>
        <Assign rdf:resource="AlignLinearSegmentReslice">
        <owl:allValuesFrom rdf:resource="AlignLinearSegmentReslice" />
        </Assign>
        <Assign rdf:resource="AlignLinearScalingTermination">
        <owl:allValuesFrom rdf:resource="AlignLinearScalingTermination" />
        </Assign>
        <Assign rdf:resource="AlignLinearSamplings">
        <owl:allValuesFrom rdf:resource="AlignLinearSamplings" />
        </Assign>
        <Assign rdf:resource="AlignLinearConvergenceThreshold">
        <owl:allValuesFrom rdf:resource="AlignLinearConvergenceThreshold" />
        </Assign>
        <Assign rdf:resource="AlignLinearRepeatedIterations">
        <owl:allValuesFrom rdf:resource="AlignLinearRepeatedIterations" />
        </Assign>
        <Assign rdf:resource="AlignLinearHaltAfterStepsWithoutImprovement">
        <owl:allValuesFrom
rdf:resource="AlignLinearHaltAfterStepsWithoutImprovement" />
        </Assign>
        <Assign rdf:resource="AlignLinearCostFunction">
```

```
        <owl:allValuesFrom rdf:resource="AlignLinearCostFunction" />
        </Assign>
    </Module>
    <Module rdf:resource="ResliceAIR">
    <rdfs:label><![CDATA[Reslice Air]]></rdfs:label>
      <GUI x="396" y="583">
      </GUI>
        <Assign rdf:resource="CombineAIRLinearTransformation">
        <owl:allValuesFrom rdf:resource="CombineAIRLinearTransformation" />
        </Assign>
        <Assign rdf:resource="ResliceAIRReslicedVolume">
        <owl:allValuesFrom rdf:resource="ResliceAIRReslicedVolume" />
        </Assign>
        <Assign rdf:resource="ResliceAIRInterpolationModel">
        <owl:allValuesFrom rdf:resource="ResliceAIRInterpolationModel" />
        </Assign>
        <Assign rdf:resource="AlignLinearScalingTermination">
        <owl:allValuesFrom rdf:resource="AlignLinearScalingTermination" />
        </Assign>
    </Module>
    <Module rdf:resource="InvertAIR">
    <rdfs:label><![CDATA[Invert Air]]></rdfs:label>
      <GUI x="175" y="327">
      </GUI>
        <Assign rdf:resource="CropTransformation">
        <owl:allValuesFrom rdf:resource="CropTransformation" />
        </Assign>
        <Assign rdf:resource="InvertAIRInvertedLinearTransformation">
        <owl:allValuesFrom
rdf:resource="InvertAIRInvertedLinearTransformation" />
        </Assign>
    </Module>
    <Module rdf:resource="CombineAIR">
    <rdfs:label><![CDATA[Combine Air]]></rdfs:label>
      <GUI x="229" y="474">
      </GUI>
        <Assign rdf:resource="CombineAIRLinearTransformation">
        <owl:allValuesFrom rdf:resource="CombineAIRLinearTransformation" />
        </Assign>
        <Assign rdf:resource="CombineAIRCLIOption2">
        <owl:allValuesFrom rdf:resource="CombineAIRCLIOption2" />
        </Assign>
        <Assign rdf:resource="InvertAIRInvertedLinearTransformation">
        <owl:allValuesFrom
rdf:resource="InvertAIRInvertedLinearTransformation" />
        </Assign>
        <Assign rdf:resource="AlignLinearLinearTransform">
        <owl:allValuesFrom rdf:resource="AlignLinearLinearTransform" />
        </Assign>
    </Module>
  </Module>

</rdf:RDF>
```

LONI Schema

http://www.loni.ucla.edu/twiki/bin/view/Pipeline/PipelineV3Syntax

```
----START---->

PipelineModules in Pipeline v3 are stored in OWL, which is a recommended
specification of the World Wide Web Consortium (W3C). THe following explains
some of the technical terms necessary for discussing pipeline module
descriptions.

Module metadata
Namespaces are defined in the document's root element
  <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
           xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
           xmlns:owl="http://www.w3.org/2002/07/owl#">

Version information
  <owl:Ontology>
    <owl:versionInfo><![CDATA[20050223]]></owl:versionInfo>
  </owl:Ontology>

IDs- Following RDF specifications, everything in a pipeline is a resource.
Every resource can be uniquely identified by its ID, specified by its
"rdf:ID" attribute. IDs are not meant to be human readable.
Resource references- In order to define relationships between resources, we
need a manner of referring to any resource from any other resource. This is
done by using the attribute "rdf:resource".
ResourceValue
Labels- Every resource can have a set of labels. These labels are intended to
be short, human readable names that identify the resource to a human. Labels
can be localized.
Comments- Every resource can have a set of comments. These comments are
intended to provide (possibly verbose) explanations of the resources
themselves. Comments can be localized.

Namespaces
Class definitions- At the top of every pipeline module file is a set of class
definitions. These class definitions define the various classes of objects to
be manipulated, e.g. File, Function, Pipeline. Definitions are very useful,
as they can denote inheritance relations. For example, SystemExecutable is a
subclass of both File and FunctionSingleton, and is written as
  <rdfs:Class rdf:ID="ExecutableFile">
    <rdfs:SubClassOf rdf:resource="File" />
    <rdfs:SubClassOf rdf:resource="FunctionSingleton" />
  </rdfs:Class>
File dependencies are another example of the usefulness of definitions. For
example, Analyze image files have a header dependency, which have the same
path as the image files, except with the file extension replacing "img" with
"hdr". This is written as
  <rdfs:Class rdf:ID="AnalyzeImageFile" extension="img">
    <Dependency rdf:resource="AnalyzeImageHeaderFile">
      <replace type="extension"><![CDATA[hdr]]></replace>
    </Dependency>
    <rdfs:SubClassOf rdf:resource="File" />
```

```
  </rdfs:Class>
```
Resource definitions- Once the classes have been defined, class instances are defined. In OWL, all objects are resources, and provided an ID, to which it can be referred. To define an Analyze image file,
```
  <AnalyzeImageFile rdf:ID="AnalyzeImageFileInstance1" />
```
Function definitions- A pipeline at its core is a sequence of Functions applied to a data set. Hence functions definitions are crucial. Since functions are resources as well, we can define functions just as easily as we define resources.
Function arguments
FunctionSingleton
FunctionPipeline
Module definitions
Module types- Currently there are 3 types of modules in the system
Pipelines- Pipelines are modules that nest other modules.
Singletons- Singletons wrap around a single Function. They refer to this function by its ID.
Breakpoints
Module argument assignments- The values of the Function arguments are populated by variable assignments.

```
   <Module rdf:resource="simplealignGrid">
      <Module rdf:resource="Crop0">
         <Assign rdf:resource="CropVolume">
            <owl:allValuesFrom rdf:resource="CropVolume" />
         </Assign>
         <Assign rdf:resource="CropPadding">
            <owl:allValuesFrom rdf:resource="CropPadding" />
         </Assign>
         <Assign rdf:resource="CropCroppedVolume">
            <owl:allValuesFrom rdf:resource="CropCroppedVolume" />
         </Assign>
         <Assign rdf:resource="CropCLIOption4">
            <owl:allValuesFrom rdf:resource="CropCLIOption4" />
         </Assign>
         <Assign rdf:resource="CropTransformation">
            <owl:allValuesFrom rdf:resource="CropTransformation" />
         </Assign>
      </Module>
   </Module>
```

The DTD of the documentions should go ->here<-

---END--->

Two links are provided in the above page, one from "FunctionSingleton" above:

----START--->

```
  <ExecutableFile rdf:ID="CropExecutableFile"
        url="pipeline://localhost//usr/local/AIR/bin/crop">
        <Argument connectionType="input" rdf:ID="CropVolume">
          <rdfs:label><![CDATA[Crop Volume]]></rdfs:label>
          <rdfs:comment><![CDATA[the name of the file to be
cropped]]></rdfs:comment>
        </Argument>
```

```
        <Argument rdf:ID="CropPadding">
          <rdfs:label><![CDATA[Crop Padding]]></rdfs:label>
          <rdfs:comment><![CDATA[the number of voxels of padding
             to add back after cropping
             (more won't be added than was cropped)]]></rdfs:comment>
        </Argument>
        <Argument connectionType="output" rdf:ID="CropCroppedVolume">
          <rdfs:label><![CDATA[Crop Cropped Volume]]></rdfs:label>
          <rdfs:comment><![CDATA[the name of the output
file]]></rdfs:comment>
        </Argument>
        <Argument rdf:ID="CropCLIOption4">
          <rdfs:label><![CDATA[Crop overwrite?(y/n)]]></rdfs:label>
          <rdfs:comment><![CDATA['y' grants permission to overwrite
output]]></rdfs:comment>
        </Argument>
        <Argument connectionType="output" rdf:ID="CropTransformation"
optional="true">
          <rdfs:label><![CDATA[Crop Transformation]]></rdfs:label>
          <rdfs:comment><![CDATA[the name of a .air file that will be created
              to describe the crop with input as the reslice file and
              output as the standard file.]]></rdfs:comment>
        </Argument>
    <rdfs:label><![CDATA[Crop executable]]></rdfs:label>
    <rdfs:comment><![CDATA[executable for Crop]]></rdfs:comment>
  </ExecutableFile>

---END--->

And a link from "FunctionPipeline" above:

----START--->

  <FunctionPipeline rdf:ID="simplealignGrid">
    <rdfs:label><![CDATA[simple align]]></rdfs:label>
    <GUI>
      <Icon rdf:resource="simplealignIcon" />
    </GUI>
    <Assign rdf:ID="Crop0">
      <owl:allValuesFrom rdf:resource="CropExecutableFile" />
    </Assign>
    <Assign rdf:ID="Crop1">
      <owl:allValuesFrom rdf:resource="CropExecutableFile" />
    </Assign>
    <Assign rdf:ID="AlignLinear">
      <owl:allValuesFrom rdf:resource="AlignLinearExecutableFile" />
    </Assign>
    <Assign rdf:ID="ResliceAIR">
      <owl:allValuesFrom rdf:resource="ResliceAIRExecutableFile" />
    </Assign>
    <Assign rdf:ID="InvertAIR">
      <owl:allValuesFrom rdf:resource="InvertAIRExecutableFile" />
    </Assign>
    <Assign rdf:ID="CombineAIR">
      <owl:allValuesFrom rdf:resource="CombineAIRExecutableFile" />
    </Assign>
  </FunctionPipeline>
```

```
---END-->
```

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>
```

BPEL4WS Schema

```
<?xml version='1.0' encoding="UTF-8"?>
<schema xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:bpws="http://schemas.xmlsoap.org/ws/2003/03/business-process/"
targetNamespace="http://schemas.xmlsoap.org/ws/2003/03/business-process/"
elementFormDefault="qualified">
<import namespace="http://schemas.xmlsoap.org/wsdl/"
schemaLocation="http://schemas.xmlsoap.org/wsdl/"/>
<complexType name="tExtensibleElements">
      <annotation>
            <documentation>
            This type is extended by other component types
            to allow elements and attributes from
            other namespaces to be added.
            </documentation>
      </annotation>
      <sequence>
            <any namespace="##other" minOccurs="0" maxOccurs="unbounded"
      processContents="lax"/>
      </sequence>
      <anyAttribute namespace="##other" processContents="lax"/>
</complexType>
<element name="process" type="bpws:tProcess"/>
<complexType name="tProcess">
      <complexContent>
            <extension base="bpws:tExtensibleElements">
                  <sequence>
```

```
                        <element name="partnerLinks" type="bpws:tPartnerLinks"
                  minOccurs="0"/>
                        <element name="partners" type="bpws:tPartners"
                  minOccurs="0"/>
                        <element name="variables" type="bpws:tVariables"
                  minOccurs="0"/>
                        <element name="correlationSets"
                  type="bpws:tCorrelationSets" minOccurs="0"/>
                        <element name="faultHandlers"
                  type="bpws:tFaultHandlers" minOccurs="0"/>
                        <element name="compensationHandler"
                  type="bpws:tCompensationHandler" minOccurs="0"/>
                        <element name="eventHandlers"
                  type="bpws:tEventHandlers" minOccurs="0"/>
                        <group ref="bpws:activity"/>
                  </sequence>
                  <attribute name="name" type="NCName" use="required"/>
                  <attribute name="targetNamespace" type="anyURI"
            use="required"/>
                        <attribute name="queryLanguage" type="anyURI"
            default="http://www.w3.org/TR/1999/REC-xpath-19991116"/>
                        <attribute name="expressionLanguage" type="anyURI"
            default="http://www.w3.org/TR/1999/REC-xpath-19991116"/>
                        <attribute name="suppressJoinFailure" type="bpws:tBoolean"
            default="no"/>
                        <attribute name="enableInstanceCompensation"
            type="bpws:tBoolean" default="no"/>
                        <attribute name="abstractProcess" type="bpws:tBoolean"
            default="no"/>
            </extension>
      </complexContent>
</complexType>
<group name="activity">
      <choice>
            <element name="empty" type="bpws:tEmpty"/>
            <element name="invoke" type="bpws:tInvoke"/>
            <element name="receive" type="bpws:tReceive"/>
            <element name="reply" type="bpws:tReply"/>
            <element name="assign" type="bpws:tAssign"/>
            <element name="wait" type="bpws:tWait"/>
            <element name="throw" type="bpws:tThrow"/>
            <element name="terminate" type="bpws:tTerminate"/>
            <element name="flow" type="bpws:tFlow"/>
            <element name="switch" type="bpws:tSwitch"/>
            <element name="while" type="bpws:tWhile"/>
            <element name="sequence" type="bpws:tSequence"/>
            <element name="pick" type="bpws:tPick"/>
            <element name="scope" type="bpws:tScope"/>
      </choice>
</group>
<complexType name="tPartnerLinks">
      <complexContent>
            <extension base="bpws:tExtensibleElements">
                  <sequence>
                        <element name="partnerLink" type="bpws:tPartnerLink"
                  minOccurs="1" maxOccurs="unbounded"/>
                  </sequence>
            </extension>
      </complexContent>
</complexType>
<complexType name="tPartnerLink">
      <complexContent>
            <extension base="bpws:tExtensibleElements">
                  <attribute name="name" type="NCName" use="required"/>
```

```
                  <attribute name="partnerLinkType" type="QName"
                  use="required"/>
                  <attribute name="myRole" type="NCName"/>
                  <attribute name="partnerRole" type="NCName"/>
            </extension>
      </complexContent>
</complexType>
<complexType name="tPartners">
      <complexContent>
            <extension base="bpws:tExtensibleElements">
                  <sequence>
                        <element name="partner" type="bpws:tPartner"
                        minOccurs="1" maxOccurs="unbounded"/>
                  </sequence>
            </extension>
      </complexContent>
</complexType>
<complexType name="tPartner">
      <complexContent>
            <extension base="bpws:tExtensibleElements">
                  <sequence>
                        <element name="partnerLink" minOccurs="1"
                        maxOccurs="unbounded">
                              <complexType>
                                    <complexContent>
                                          <extension
                                    base="bpws:tExtensibleElements">
                                                <attribute name="name"
                                                type="NCName"
                                                use="required"/>
                                          </extension>
                                    </complexContent>
                              </complexType>
                        </element>
                  </sequence>
                  <attribute name="name" type="NCName" use="required"/>
            </extension>
      </complexContent>
</complexType>
<complexType name="tFaultHandlers">
      <complexContent>
            <extension base="bpws:tExtensibleElements">
                  <sequence>
                        <element name="catch" type="bpws:tCatch"
                        minOccurs="0" maxOccurs="unbounded"/>
                        <element name="catchAll"
                        type="bpws:tActivityOrCompensateContainer"
                        minOccurs="0"/>
                  </sequence>
            </extension>
      </complexContent>
</complexType>
<complexType name="tCatch">
      <complexContent>
            <extension base="bpws:tActivityOrCompensateContainer">
                  <attribute name="faultName" type="QName"/>
                  <attribute name="faultVariable" type="NCName"/>
            </extension>
      </complexContent>
</complexType>
<complexType name="tActivityContainer">
      <complexContent>
            <extension base="bpws:tExtensibleElements">
                  <sequence>
```

```
                        <group ref="bpws:activity"/>
                  </sequence>
            </extension>
      </complexContent>
</complexType>
<complexType name="tActivityOrCompensateContainer">
      <complexContent>
            <extension base="bpws:tExtensibleElements">
                  <choice>
                        <group ref="bpws:activity"/>
                        <element name="compensate" type="bpws:tCompensate"/>
                  </choice>
            </extension>
      </complexContent>
</complexType>
<complexType name="tEventHandlers">
      <complexContent>
            <extension base="bpws:tExtensibleElements">
                  <sequence>
                        <element name="onMessage" type="bpws:tOnMessage"
                        minOccurs="0" maxOccurs="unbounded"/>
                        <element name="onAlarm" type="bpws:tOnAlarm"
                        minOccurs="0" maxOccurs="unbounded"/>
                  </sequence>
            </extension>
      </complexContent>
</complexType>
<complexType name="tOnMessage">
      <complexContent>
            <extension base="bpws:tExtensibleElements">
                  <sequence>
                        <element name="correlations" type="bpws:tCorrelations"
                        minOccurs="0"/>
                        <group ref="bpws:activity"/>
                  </sequence>
                  <attribute name="partnerLink" type="NCName" use="required"/>
                  <attribute name="portType" type="QName" use="required"/>
                  <attribute name="operation" type="NCName" use="required"/>
                  <attribute name="variable" type="NCName"
                  use="optional"/>
            </extension>
      </complexContent>
</complexType>
<complexType name="tOnAlarm">
      <complexContent>
            <extension base="bpws:tActivityContainer">
                  <attribute name="for" type="bpws:tDuration-expr"/>
                  <attribute name="until" type="bpws:tDeadline-expr"/>
            </extension>
      </complexContent>
</complexType>
<complexType name="tCompensationHandler">
      <complexContent>
            <extension base="bpws:tActivityOrCompensateContainer"/>
      </complexContent>
</complexType>
<complexType name="tVariables">
      <complexContent>
            <extension base="bpws:tExtensibleElements">
                  <sequence>
                        <element name="variable"
                        type="bpws:tVariable"
                        maxOccurs="unbounded"/>
                  </sequence>
```

```
            </extension>
      </complexContent>
</complexType>
<complexType name="tVariable">
<!-- variable does not allow extensibility elements
because otherwise its content model would be non-deterministic -->
      <attribute name="name" type="NCName" use="required"/>
      <attribute name="messageType" type="QName" use = “optional”/>
      <attribute name="type" type="QName" use = “optional”/>
      <attribute name="element" type="QName" use = “optional”/>
      <anyAttribute namespace="##other" processContents="lax"/>
</complexType>
<complexType name="tCorrelationSets">
      <complexContent>
            <extension base="bpws:tExtensibleElements">
                  <sequence>
                        <element name="correlationSet"
                        type="bpws:tCorrelationSet"
                        maxOccurs="unbounded"/>
                  </sequence>
            </extension>
      </complexContent>
</complexType>
<complexType name="tCorrelationSet">
      <complexContent>
            <extension base="bpws:tExtensibleElements">
                  <attribute name="properties" use="required">
                        <simpleType>
                              <list itemType="QName"/>
                        </simpleType>
                  </attribute>
                  <attribute name="name" type="NCName" use="required"/>
            </extension>
      </complexContent>
</complexType>
<complexType name="tActivity">
      <complexContent>
            <extension base="bpws:tExtensibleElements">
                  <sequence>
                        <element name="target" type="bpws:tTarget"
                        minOccurs="0" maxOccurs="unbounded"/>
                        <element name="source" type="bpws:tSource"
                        minOccurs="0" maxOccurs="unbounded"/>
                  </sequence>
                  <attribute name="name" type="NCName"/>
                  <attribute name="joinCondition"
                  type="bpws:tBoolean-expr"/>
                  <attribute name="suppressJoinFailure"
                  type="bpws:tBoolean" default="no"/>
            </extension>
      </complexContent>
</complexType>
<complexType name="tSource">
      <complexContent>
            <extension base="bpws:tExtensibleElements">
                  <attribute name="linkName" type="NCName" use="required"/>
                  <attribute name="transitionCondition"
                  type="bpws:tBoolean-expr"/>
            </extension>
      </complexContent>
</complexType>
<complexType name="tTarget">
      <complexContent>
            <extension base="bpws:tExtensibleElements">
```

```
                  <attribute name="linkName" type="NCName" use="required"/>
            </extension>
      </complexContent>
</complexType>
<complexType name="tEmpty">
      <complexContent>
            <extension base="bpws:tActivity"/>
      </complexContent>
</complexType>
<complexType name="tCorrelations">
      <complexContent>
            <extension base="bpws:tExtensibleElements">
                  <sequence>
                        <element name="correlation" type="bpws:tCorrelation"
                        minOccurs="1" maxOccurs="unbounded" />
                  </sequence>
            </extension>
      </complexContent>
</complexType>
<complexType name="tCorrelation">
      <complexContent>
            <extension base="bpws:tExtensibleElements">
                  <attribute name="set" type="NCName" use="required"/>
                  <attribute name="initiate" type="bpws:tBoolean"
                  default="no"/>
            </extension>
      </complexContent>
</complexType>
<complexType name="tCorrelationsWithPattern">
      <complexContent>
            <extension base="bpws:tExtensibleElements">
                  <sequence>
                        <element name="correlation"
                        type="bpws:tCorrelationWithPattern"
                        minOccurs="1"
                        maxOccurs="unbounded"/>
                  </sequence>
            </extension>
      </complexContent>
</complexType>
<complexType name="tCorrelationWithPattern">
      <complexContent>
            <extension base="bpws:tCorrelation">
                  <attribute name="pattern">
                        <simpleType>
                              <restriction base="string">
                                    <enumeration value="in" />
                                    <enumeration value="out" />
                                    <enumeration value="out-in" />
                              </restriction>
                        </simpleType>
                  </attribute>
            </extension>
      </complexContent>
</complexType>
<complexType name="tInvoke">
      <complexContent>
            <extension base="bpws:tActivity">
                  <sequence>
                        <element name="correlations"
                        type="bpws:tCorrelationsWithPattern"
                        minOccurs="0" maxOccurs="1"/>
                        <element name="catch" type="bpws:tCatch"
                        minOccurs="0" maxOccurs="unbounded"/>
```

```
                        <element name="catchAll"
                        type="bpws:tActivityOrCompensateContainer"
                        minOccurs="0"/>
                        <element name="compensationHandler"
                        type="bpws:tCompensationHandler" minOccurs="0"/>
                  </sequence>
                  <attribute name="partnerLink" type="NCName" use="required"/>
                  <attribute name="portType" type="QName" use="required"/>
                  <attribute name="operation" type="NCName" use="required"/>
                  <attribute name="inputVariable"
                  type="NCName" use="optional"/>
                  <attribute name="outputVariable" type="NCName"
                  use="optional"/>
            </extension>
      </complexContent>
</complexType>
<complexType name="tReceive">
      <complexContent>
            <extension base="bpws:tActivity">
                  <sequence>
                        <element name="correlations"
                        type="bpws:tCorrelations" minOccurs="0"/>
                  </sequence>
                  <attribute name="partnerLink" type="NCName" use="required"/>
                  <attribute name="portType" type="QName" use="required"/>
                  <attribute name="operation" type="NCName" use="required"/>
                  <attribute name="variable" type="NCName" use="optional"/>
                  <attribute name="createInstance" type="bpws:tBoolean"
                  default="no"/>
            </extension>
      </complexContent>
</complexType>
<complexType name="tReply">
      <complexContent>
            <extension base="bpws:tActivity">
                  <sequence>
                        <element name="correlations"
                        type="bpws:tCorrelations" minOccurs="0"/>
                  </sequence>
                  <attribute name="partnerLink" type="NCName" use="required"/>
                  <attribute name="portType" type="QName" use="required"/>
                  <attribute name="operation" type="NCName" use="required"/>
                  <attribute name="variable" type="NCName"
                  use="optional"/>
                  <attribute name="faultName" type="QName"/>
            </extension>
      </complexContent>
</complexType>
<complexType name="tAssign">
      <complexContent>
            <extension base="bpws:tActivity">
                  <sequence>
                        <element name="copy" type="bpws:tCopy"
                        minOccurs="1" maxOccurs="unbounded"/>
                  </sequence>
            </extension>
      </complexContent>
</complexType>
<complexType name="tCopy">
      <complexContent>
            <extension base="bpws:tExtensibleElements">
                  <sequence>
                        <element ref="bpws:from"/>
                        <element ref="bpws:to"/>
```

```
                </sequence>
            </extension>
        </complexContent>
</complexType>
<element name="from" type="bpws:tFrom"/>
<complexType name="tFrom">
        <complexContent>
            <extension base="bpws:tExtensibleElements">
                <attribute name="variable" type="NCName"/>
                <attribute name="part" type="NCName"/>
                <attribute name="query" type="string"/>
                <attribute name="property" type="QName"/>
                <attribute name="partnerLink" type="NCName"/>
                <attribute name="endpointReference" type="bpws:tRoles"/>
                <attribute name="expression" type="string"/>
                <attribute name="opaque" type="bpws:tBoolean"/>
            </extension>
        </complexContent>
</complexType>
<element name="to">
        <complexType>
            <complexContent>
                <restriction base="bpws:tFrom">
                    <attribute name="expression" type="string"
                    use="prohibited"/>
                    <attribute name="opaque" type="bpws:tBoolean"
                    use="prohibited"/>
                    <attribute name="endpointReference" type="bpws:tRoles"
                    use="prohibited"/>
                </restriction>
            </complexContent>
        </complexType>
</element>
<complexType name="tWait">
        <complexContent>
            <extension base="bpws:tActivity">
                <attribute name="for"
                type="bpws:tDuration-expr"/>
                <attribute name="until"
                type="bpws:tDeadline-expr"/>
            </extension>
        </complexContent>
</complexType>
<complexType name="tThrow">
        <complexContent>
            <extension base="bpws:tActivity">
                <attribute name="faultName" type="QName" use="required"/>
                <attribute name="faultVariable" type="NCName"/>
            </extension>
        </complexContent>
</complexType>
<complexType name="tCompensate">
        <complexContent>
            <extension base="bpws:tActivity">
                <attribute name="scope" type="NCName"/>
            </extension>
        </complexContent>
</complexType>
<complexType name="tTerminate">
        <complexContent>
            <extension base="bpws:tActivity"/>
        </complexContent>
</complexType>
<complexType name="tFlow">
```

```
      <complexContent>
            <extension base="bpws:tActivity">
                  <sequence>
                        <element name="links" type="bpws:tLinks"
                        minOccurs="0"/>
                        <group ref="bpws:activity" maxOccurs="unbounded"/>
                  </sequence>
            </extension>
      </complexContent>
</complexType>
<complexType name="tLinks">
      <complexContent>
            <extension base="bpws:tExtensibleElements">
                  <sequence>
                        <element name="link"
                        type="bpws:tLink"
                        maxOccurs="unbounded"/>
                  </sequence>
            </extension>
      </complexContent>
</complexType>
<complexType name="tLink">
      <complexContent>
            <extension base="bpws:tExtensibleElements">
                  <attribute name="name" type="NCName" use="required"/>
            </extension>
      </complexContent>
</complexType>
<complexType name="tSwitch">
      <complexContent>
            <extension base="bpws:tActivity">
                  <sequence>
                        <element name="case" maxOccurs="unbounded">
                              <complexType>
                                    <complexContent>
                                          <extension
                                          base="bpws:tActivityContainer">
                                                <attribute name="condition"
                                                type="bpws:tBoolean-expr"
                                                use="required"/>
                                          </extension>
                                    </complexContent>
                              </complexType>
                        </element>
                        <element name="otherwise"
                        type="bpws:tActivityContainer"
                        minOccurs="0"/>
                  </sequence>
            </extension>
      </complexContent>
</complexType>
<complexType name="tWhile">
      <complexContent>
            <extension base="bpws:tActivity">
                  <sequence>
                        <group ref="bpws:activity"/>
                  </sequence>
                  <attribute name="condition"
                  type="bpws:tBoolean-expr"
                  use="required"/>
            </extension>
      </complexContent>
</complexType>
<complexType name="tSequence">
```

```
      <complexContent>
            <extension base="bpws:tActivity">
                  <sequence>
                        <group ref="bpws:activity" maxOccurs="unbounded"/>
                  </sequence>
            </extension>
      </complexContent>
</complexType>
<complexType name="tPick">
      <complexContent>
            <extension base="bpws:tActivity">
                  <sequence>
                        <element name="onMessage"
                        type="bpws:tOnMessage"
                        maxOccurs="unbounded"/>
                        <element name="onAlarm"
                        type="bpws:tOnAlarm" minOccurs="0"
                        maxOccurs="unbounded"/>
                  </sequence>
                  <attribute name="createInstance"
                  type="bpws:tBoolean" default="no"/>
            </extension>
      </complexContent>
</complexType>
<complexType name="tScope">
      <complexContent>
            <extension base="bpws:tActivity">
                  <sequence>
                        <element name="variables"
                        type="bpws:tVariables"
                        minOccurs="0"/>
                        <element name="correlationSets"
                        type="bpws:tCorrelationSets"
                        minOccurs="0"/>
                        <element name="faultHandlers"
                        type="bpws:tFaultHandlers"
                        minOccurs="0"/>
                        <element name="compensationHandler"
                        type="bpws:tCompensationHandler"
                        minOccurs="0"/>
                        <element name="eventHandlers"
                        type="bpws:tEventHandlers"
                        minOccurs="0"/>
                        <group ref="bpws:activity"/>
                  </sequence>
                  <attribute name="variableAccessSerializable"
                  type="bpws:tBoolean"
                  default="no"/>
            </extension>
      </complexContent>
</complexType>
<simpleType name="tBoolean-expr">
      <restriction base="string"/>
</simpleType>
<simpleType name="tDuration-expr">
      <restriction base="string"/>
</simpleType>
<simpleType name="tDeadline-expr">
      <restriction base="string"/>
</simpleType>
<simpleType name="tBoolean">
      <restriction base="string">
            <enumeration value="yes"/>
            <enumeration value="no"/>
```

```
      </restriction>
</simpleType>
<simpleType name="tRoles">
      <restriction base="string">
            <enumeration value="myRole"/>
            <enumeration value="partnerRole"/>
      </restriction>
</simpleType>
</schema>
```

Partner Link Type Schema

```
<?xml version='1.0' encoding="UTF-8"?>
<schema xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:plnk="http://schemas.xmlsoap.org/ws/2003/05/partner-link/"
targetNamespace="http://schemas.xmlsoap.org/ws/2003/05/partner-link/"
elementFormDefault="qualified">
      <element name="partnerLinkType" type="plnk:tPartnerLinkType"/>
      <complexType name="tPartnerLinkType">
            <sequence>
                  <element name="role" type="plnk:tRole" minOccurs="1"
            maxOccurs="2"/>
            </sequence>
            <attribute name="name" type="NCName" use="required"/>
      </complexType>
      <complexType name="tRole">
            <sequence>
                  <element name="portType" minOccurs="1" maxOccurs="1">
                        <complexType>
                              <attribute name="name" type="QName"
                              use="required"/>
                        </complexType>
                  </element>
            </sequence>
            <attribute name="name" type="NCName" use="required"/>
      </complexType>
</schema>
```

Message Properties Schema

```
<?xml version='1.0' encoding="UTF-8"?>
<schema xmlns="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://schemas.xmlsoap.org/ws/2003/03/business-process/"
xmlns:wsbp="http://schemas.xmlsoap.org/ws/2003/03/business-process/"
elementFormDefault="qualified">
      <element name="property">
            <complexType>
                  <attribute name="name" type="NCName" use="required"/>
                  <attribute name="type" type="QName" use="required"/>
            </complexType>
      </element>
      <element name="propertyAlias">
            <complexType>
                  <attribute name="propertyName" type="QName" use="required"/>
                  <attribute name="messageType" type="QName" use="required"/>
                  <attribute name="part" type="NCName"/>
                  <attribute name="query" type="string"/>
            </complexType>
      </element>
</schema>
```