JWT –
Java Workflow Tooling

Title of the document

Comparison of Business Process Metamodels

Document information

last changes	version
12.03.2007	1.0

Document created by

Florian Lautenbacher

Address

Programming Distributed Systems Lab
Prof. Dr. Bernhard Bauer
Institute of Computer Science
University of Augsburg
Universitaetsstrasse 14
D-86135 Augsburg
Germany
phone: +49 821 598 2174
mail: lautenbacher@ds-lab.org
www: www.ds-lab.org

Content

Images

1 Summary

In the project JWT (Java Workflow Tooling, which is an Eclipse Technology Subproject) we envision the usage of a modeling tool for several workflow engines. This modeling tool should be based on a clearly defined meta-model. Therefore, we compare existing meta-models to get an overview which parts exist and are important for workflow execution. We put a special emphasis on the meta-model of AgilPro which will be the starting point for JWT and try to seek which elements are still missing.
In an upcoming version we will also compare these meta-models with the languages XPDL and WS-BPEL, where the former will probably be generated in the context of JWT and the latter should at least be possible to generate.

2 Existing Business Process Metamodels

2.1 Simple Business Process Metamodel

Figure 1 shows a simple meta-model for business processes (taken from [2]). Each process contains activities which are connected. An activity could be a link to a subprocess, a route for controlling the workflow, an application activity concerning a resource or a service. Each process contains data that are transmitted from one activity to another. On the instance side each process has one instance and each activity, too. The data gets a value in each process and the activity instances can have different states.

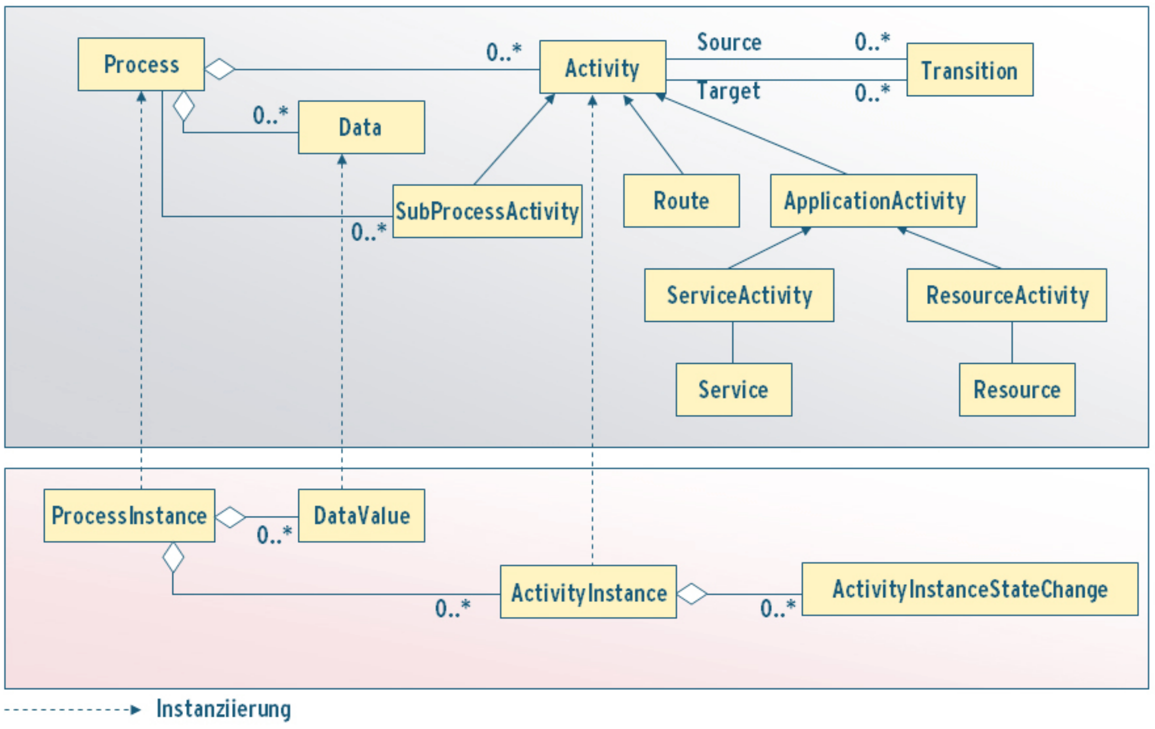

Figure 1: Simple meta-model for business processes [2]

2.2 BPDM Metamodel

[3] describes the final submission for the Business Process Definition Metamodel (BPDM) of the OMG. It represents business process models independently of the modeling notation and provides a robust serialization mechanism. It defines a shared vocabulary for process modeling concepts distinguishing complementary views of a process:

- orchestration includes the traditional view where sequences of activities are carried out
- choreography is a more abstract notion and describes interactions of entities each of which may have their own internal orchestration processes.

The activity oriented view is what the business process does, the interaction view is the definition of the commitments made by the parties. BPDM uses the terms "Process" and "Interaction Protocol" instead of orchestration and choreography in order to avoid misinterpretation. Figure 2 shows an overview about the structure of BPDM.

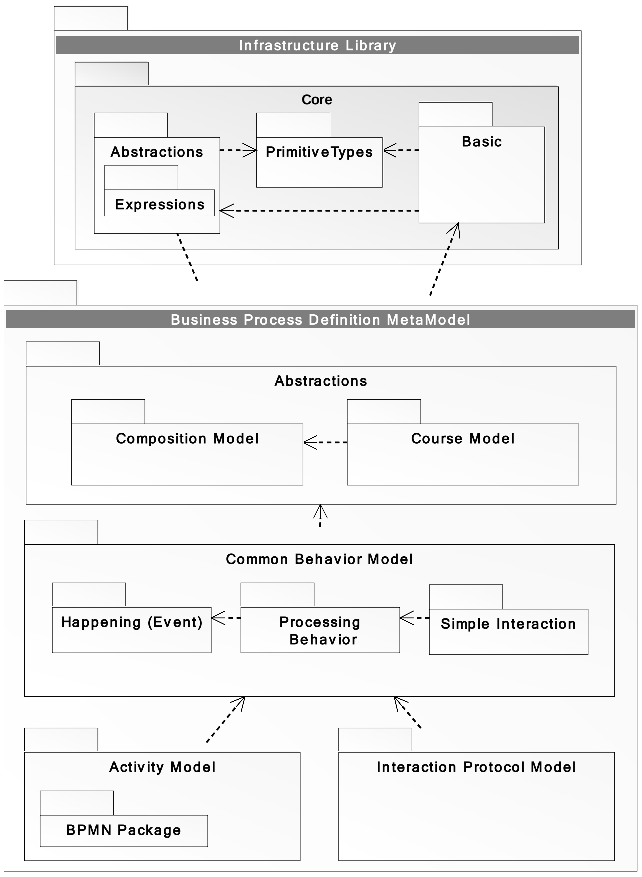

Figure 2: BPDM Structure

As one can see in Figure 3 (which is only a very small part of the BPDM) each process consists of process steps which can be activities like simple activities, sub-process activities which like to other processes or embedded processes.

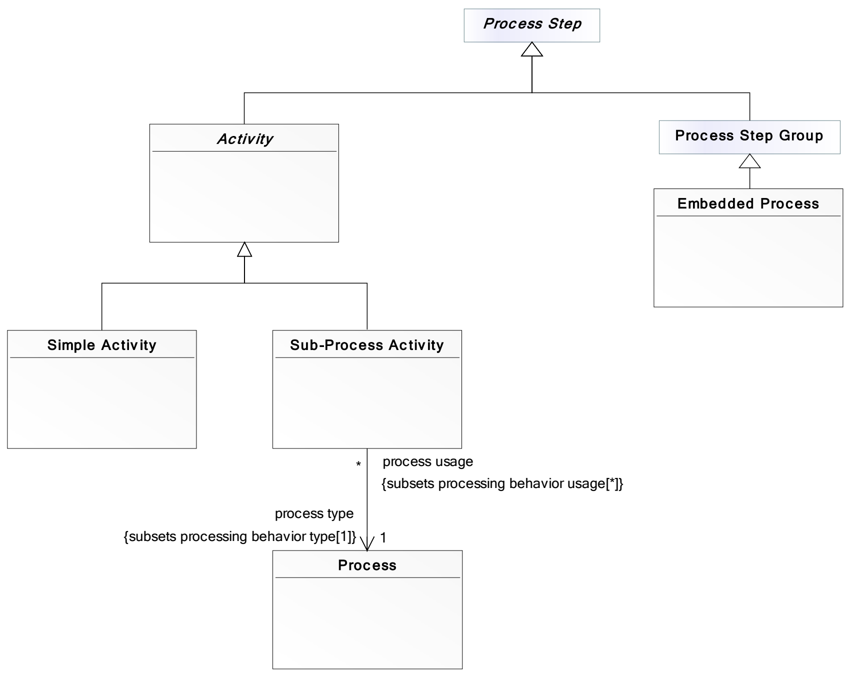

Figure 3: BPDM Metamodel: Activities

2.3 Metamodel of AgilPro / Eclipse JWT

This section describes the meta-model of Eclipse Java Workflow Tooling (JWT) and the underlying AgilPro contribution (see www.agilpro.eu or www.eclipse.org/jwt). The meta-model consists of several packages which are based on each other. The first one describes the graphical constraints whereas the latter ones are for the "real" meta-model concepts.

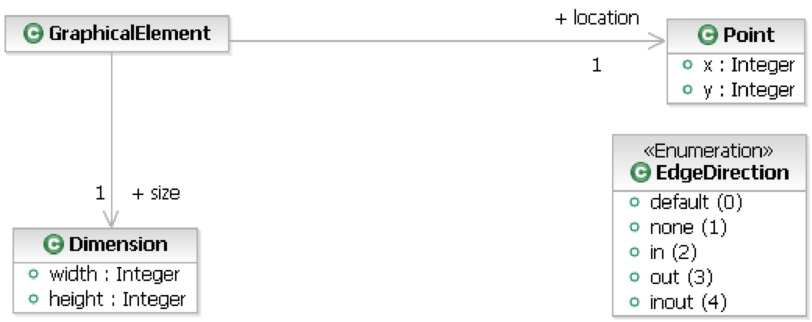

Figure 4: JWT – View

Each element which is visible in the graphical pane is a *GraphicalElement*. This has a location called *Point* with x and y value as well as a size (*Dimension*) specifying the width and height of the element. Additional there is the concept of an *EdgeDirection* which specifies whether an edge has arrows on one, both or none ends.

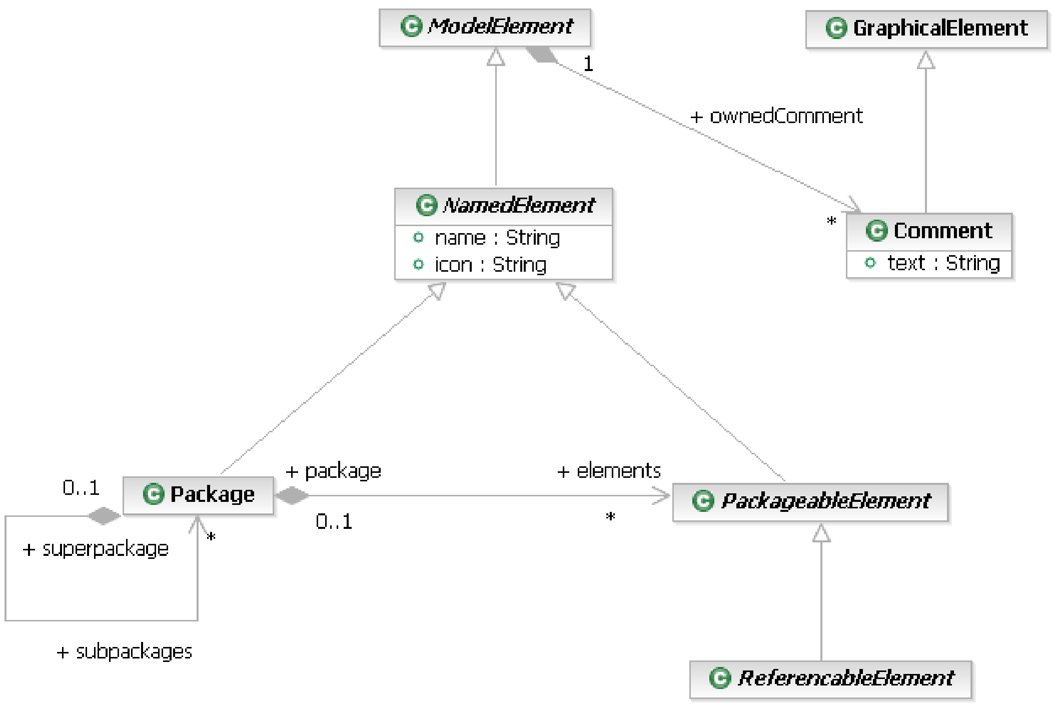

Figure 5: JWT – Core

Every element in JWT is a *ModelElement*. A *ModelElement* is the basic unit and the most abstract element of our meta-model. Every model element can have a textual *Comment*. A special kind of a model element is a *NamedElement*. All elements that have a name and optional an icon are at least *NamedElements*. A Package is a *NamedElement* and can have subpackages or other *PackageableElements*. This enables the user to structure his/her processes that belong to a specific area or to structure other elements that belong somehow together. A *ReferencableElement* is an element that can be packaged and referenced by other other elements (so called *References* introduced later).

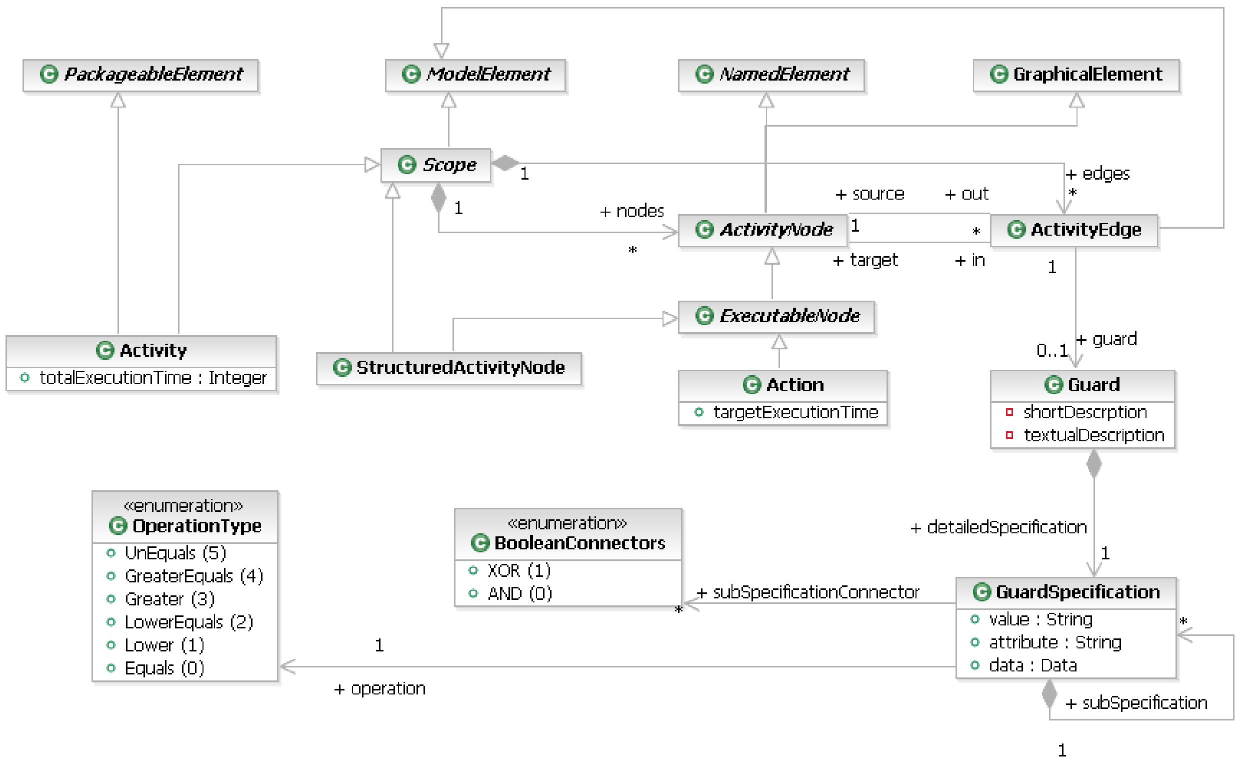

Figure 6: JWT – processes

All processes modelled with Eclipse JWT are *Activities*. An activity is a *PackageableElement* and can therefore be structured in packages. An *Activity* is a subclass of *Scope* which includes all elements in a graphical model. Examples for those elements are *ActivityNodes* and *ActivityEdges*. One example for an *ActivityNode* is an *Action* which is executable (subclass of *ExecutableNode*) and has a name and optional an icon (subclass of *NamedElement*). A *StructuredActivityNode* contains as an own scope itself *ActivityNodes* and *ActivityEdges*, but is itself executable from other nodes, too. Each *ActivityEdge* connects two *ActivityNodes* and might be constrained with a *Guard* which has a textualDescription and a more detailedDescription which can be simple Boolean terms (using the *OperationType*) or more complex terms connected through *BooleanConnectors*. Using the parameters of *Activity* (totalExecutionTime) and of all *Action*s (targetExecutionTime) one can simulate the duration of the process and compare it with the predefined value.

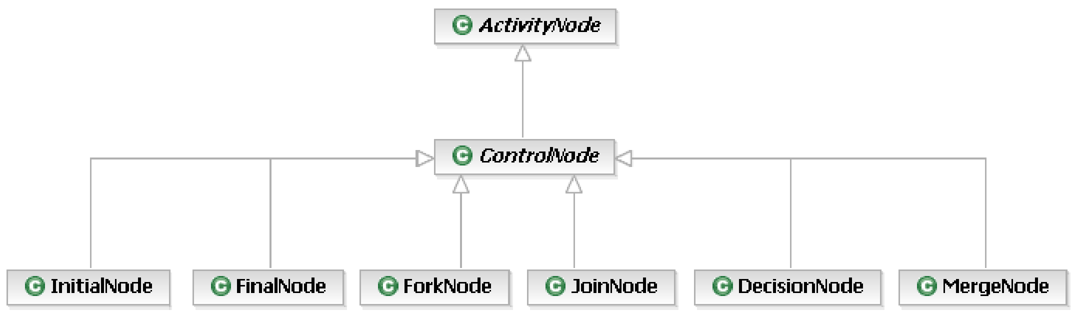

Figure 7: JWT – control nodes

To model the flow of several *ExecutableNode*s one can use *ControlNode*s. To model the start or finish of a process the *InitialNode* and *FinalNode* can be applied. To model parallel process flows and the synchronization afterwards one can use the *ForkNode* or the *JoinNode* respectively. For exclusive choices and merges afterwards the *DecisionNode* and *MergeNode* are available to the modeller.

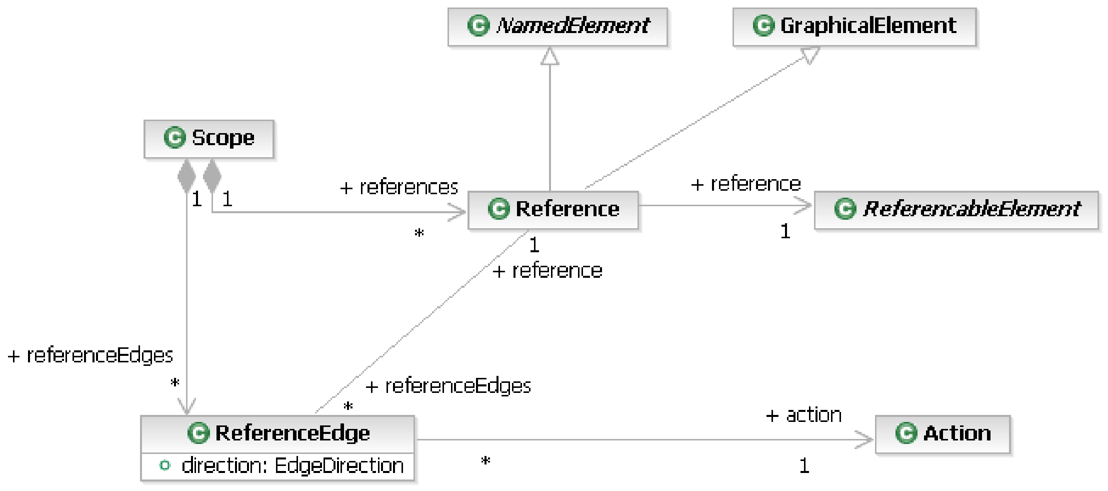

Figure 8: JWT – references

To include elements into the current activity that are normally outside the scope and defined for more than one process model, one can use the *Reference* to point to an existing *ReferenceableElement*. These References can be connected through *ReferenceEdges* with Actions. Example for a *ReferenceableElement* would be a *Role*, an *Application*, *Data*, etc. as shown later.

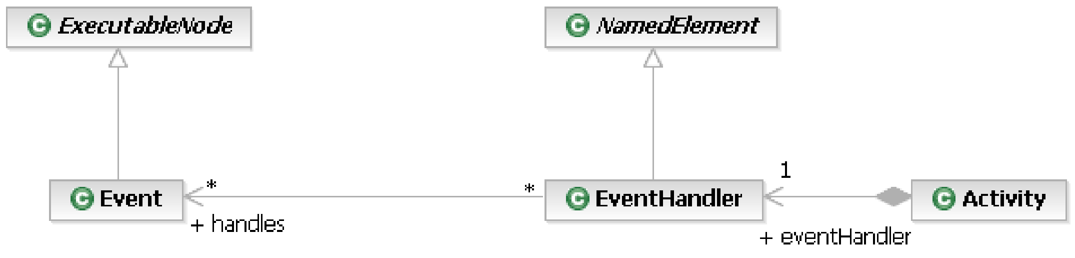

Figure 9: JWT – Events

To have the possibility to react to events from outside, one can include an *Event* into the process model. An *Event* is an *ExecutableNode* (similar to an *Action*). Each *Activity* includes an *EventHandler* who is responsible for the handling of an occurred *Event*. Such an event could be the arrival of a message, a time-out, etc.

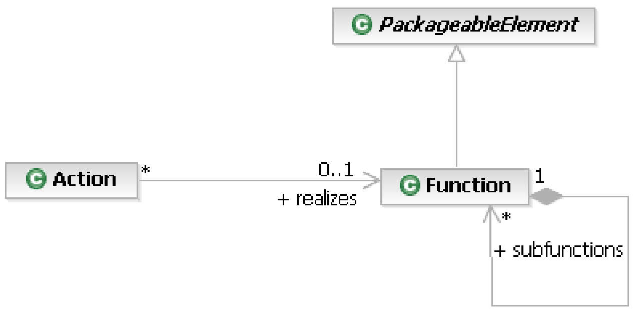

Figure 10: JWT – Functions

Each *Action* can be clustered into specific *Functions*. A function describes the kind of an action (e.g. Accounting). Each *Function* can be include in packages and might have several sub-functions belonging to itself.

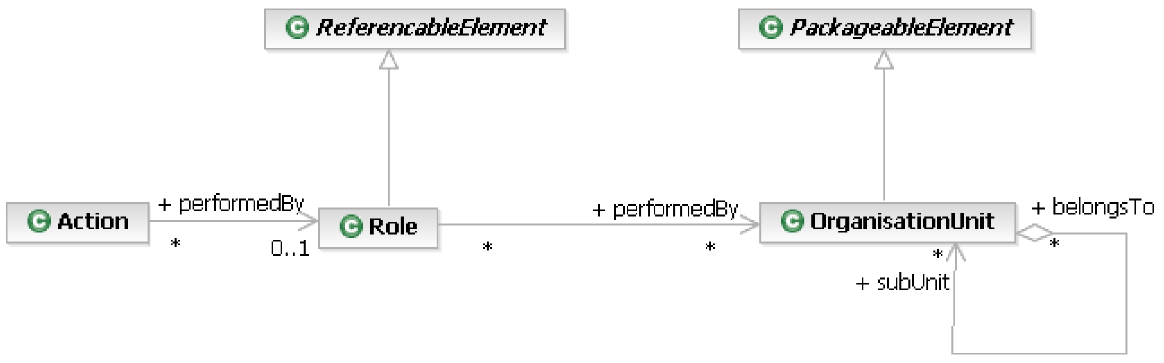

Figure 11: JWT – Organisations

Each *Action* can be performed either automatically or by a specific *Role* of an Organization. *Roles* are defined not only for one process model, but for all processes and are therefore *ReferencableElements*. *Roles* can be grouped in *OrganisationUnits* which themselves can have sub units, too.

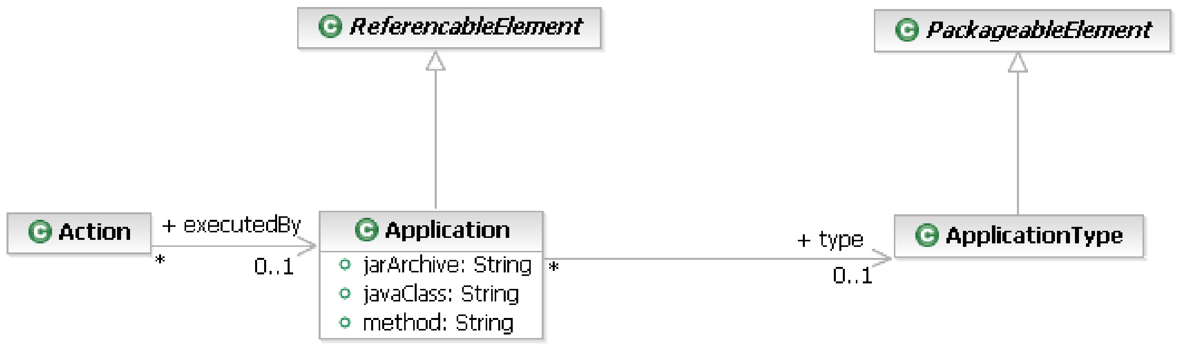

Figure 12: JWT – Applications

Each *Action* can be executed manually or alternatively by specific applications of the IT system. Again, *Applications* are defined for all kind of models and are therefore *ReferencableElements*. Each *Application* can have an *ApplicationType* which clusters the applications. An application can be specified describing the javaClass and method which should be invocated and in which jarArchive this class is.

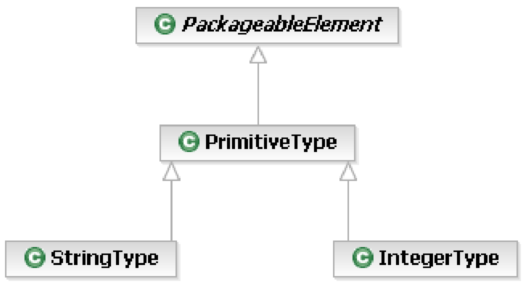

Figure 13: JWT – Primitive Types

An *Application* needs input and output data for its execution. These could either be *PrimitiveTypes* like textual *StringTypes* or numerical *IntegerTypes* or more complex types.

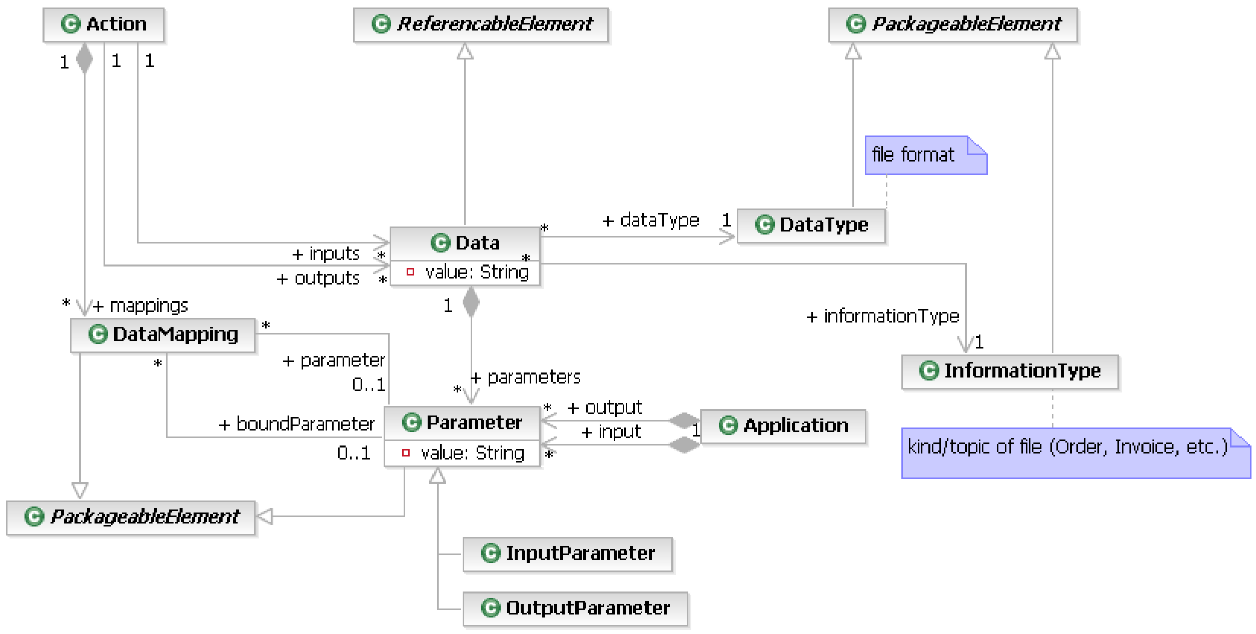

Figure 14: JWT – Data

Complex *Data* types can be described using their *DataType* which says something about the file format: is it a simple text file, an XML-file, an Excel sheet, a Word document, etc. On the other side it is possible to describe the *InformationType*, e.g. whether this is an order, an invoice, and so on. *Actions* either need these *Data* for their execution (inputs) or produce them after execution (outputs). Each action can consist of several parts, called parameters. Similar, applications can have parameters for their execution. To bind these parameters together the *DataMapping* exists which belongs to an *Action*.

2.4 The Unified Modeling Language (UML)

The Unified Modeling Language (UML) [4,5] is a standard modeling language for visualizing (using the standardized graphic UML notations) and specifying the static structure, dynamic behavior and model organization. UML consists of a notation for describing the syntax of the modeling language and a graphical notation and a meta-model which describes the static semantics of UML. The UML specification consists of the Infrastructure which defines foundational language constructs required for UML and of the Superstructure which defines user level constructs (diagrams).
UML offers the modeling of 13 different types of diagrams, six for the modeling of system structures and details of the static system and seven diagrams to model the dynamic behavior of a system.

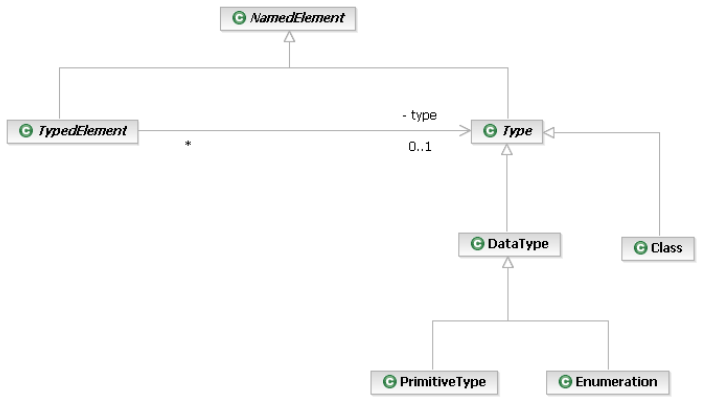

Figure 15: UML Core: type system

The UML Core describes elements that are needed in most of the other packages defined by the UML-Superstructure. An important feature is to have elements with a type, which are called TypedElements in UML and where type can either be a DataType or e.g. a Class (especially important in Class Diagrams for the modeling of software systems).

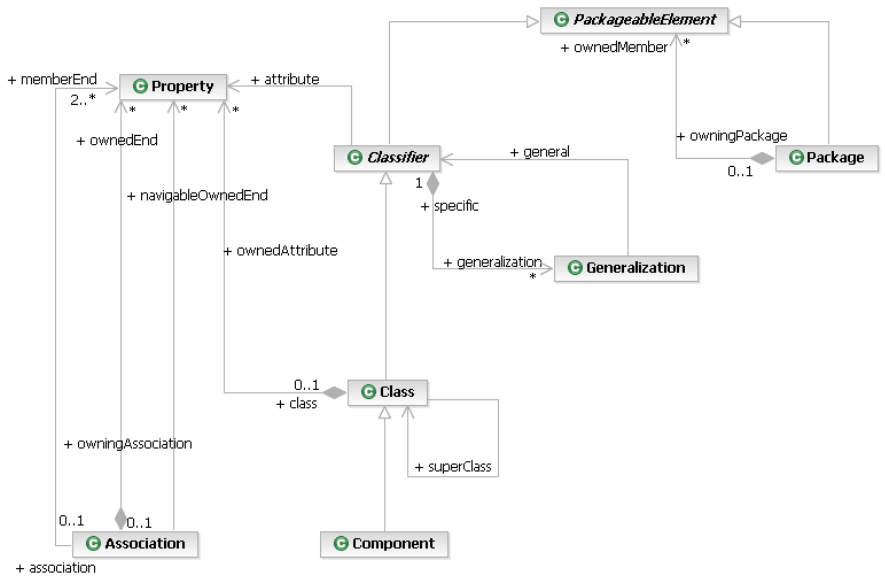

Figure 16: UML Packages, Classes, Properties and Associations

The mostly used UML-elements and graphical notation are defined in the Classes package. Class diagrams are widely common in software methodologies and are used in the analysis (e.g. conceptual modeling of the domain) and design phase (platform independent description as well as platform specific description) to describe classes and interfaces with their attributes, operations and associations (including aggregation and composition), but also generalization and dependencies among them. Class diagrams can be used for the definition of organizational models; in particular the static aspects of the organizations, its associations and part-of relationships can be modeled.

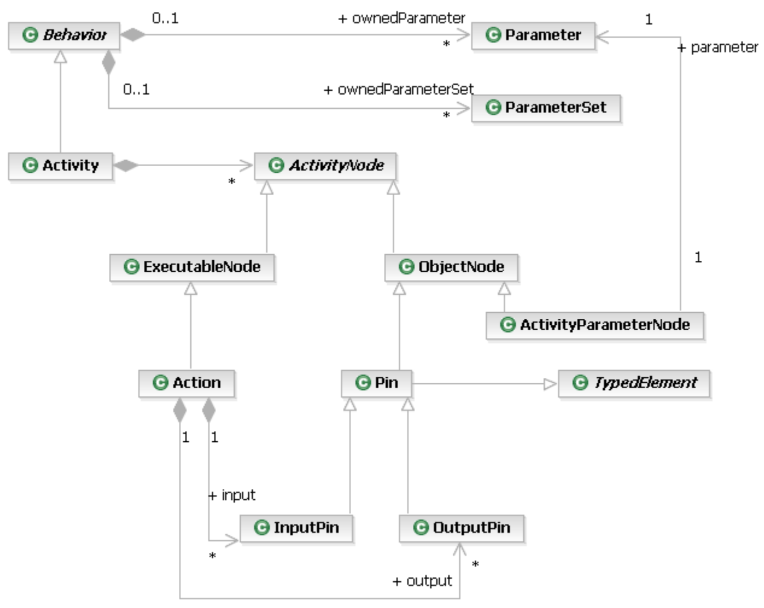

Figure 17: UML Activity: Activity nodes

In the Activity-packages of UML the basic concepts for modeling a process flow are defined. Activity modeling emphasizes the sequence and conditions for coordinating lower-level behaviors. The actions coordinated by activity models can be initiated because other actions finish executing, because objects and data become available or because events occur external to the flow. Each action can have inputs and outputs similar to the whole activity which can have parameters which can be grouped by parameter sets.

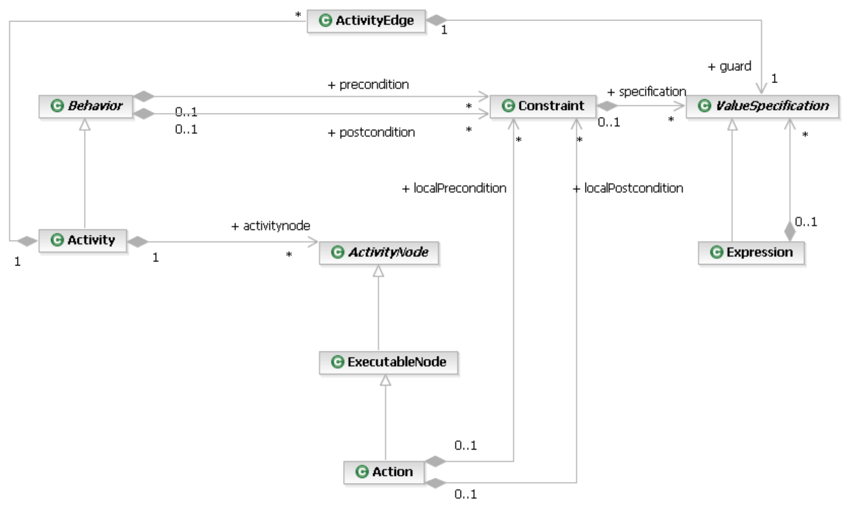

Figure 18: UML Activity: Constraints

One can specify constraints, such as preconditions and effects, on an action as well as on the whole activity. These constraints include expressions in a language such as the Object Constraint Language (OCL).

2.5 Event-driven Process Chains (EPC)

Event-driven Process Chains are a method developed by Scheer, Keller and Nüttgens within the framework of Architecture of Integrated Information Systems (ARIS) to model business processes. The ARIS concept involves dividing complex business processes into separate views and integrating these separate views to form a complete overview about one business process. These are

- the function view which describes the activities within a company which are to be performed, the enumeration of the individual subfunctions that belong to the overall relationships and the relationships between the functions
- the data view which contains events and status information. Events are created by processing functions or by actors outside of the model. An event may act as a pre- or postcondition of a function.
- the organization view for the structure and relationships between users and organization units which are responsible for performing a function and
- the resource view which includes general conditions for describing other components and deliverables that represent services or products that functions produce or need.

The meta-model of Event-driven Process Chains in Figure 19 shows the most important elements of EPCs.

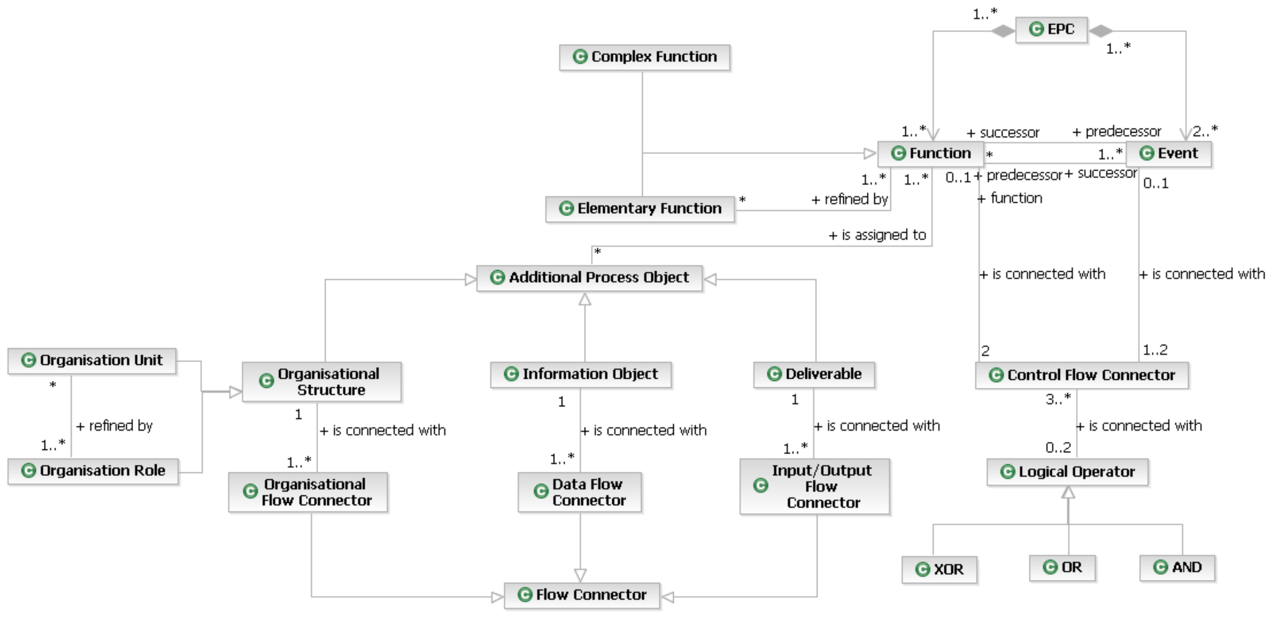

Figure 19: EPC metamodel (analogue to [6])

2.6 Generic meta-model by List/Korherr

In [7] the authors evaluate seven existing business process modeling languages using a newly developed generic meta-model for business processes. This meta-model consists of five perspectives: the four perspectives of Curtis et al. [8] named organisational, functional, behavioural and informational perspective and an own perspective called business process context perspective which provides an overview perspective of the process and describes major business process characteristics such as goals and their measures, the deliverables, the process owner, the process type and the customer at a glance (cf. Figure 20).

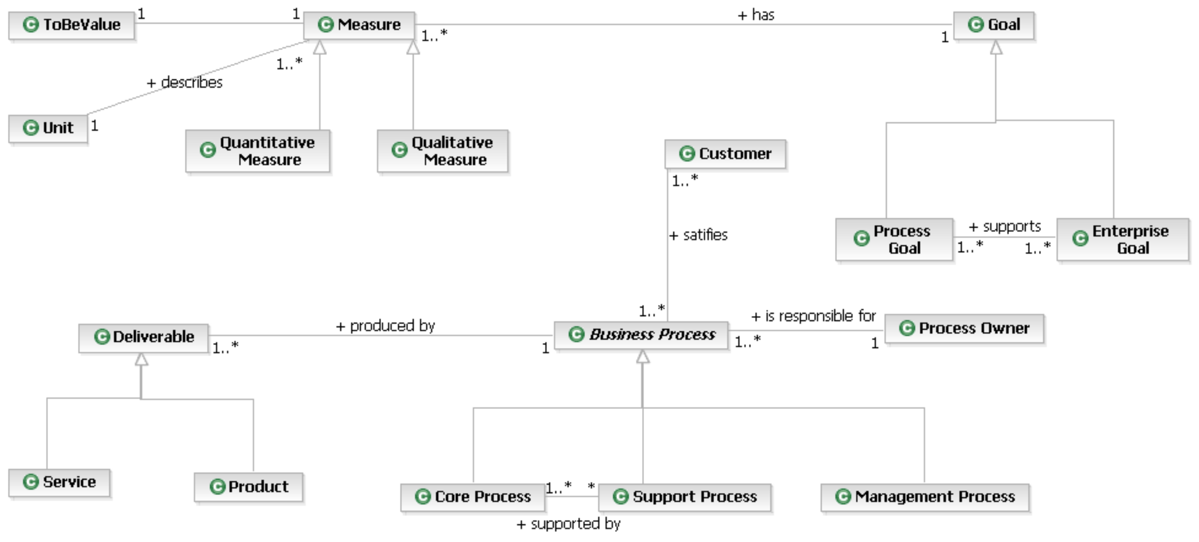

Figure 20: List/Korherr: Business Process Context Perspective

The functional perspective (Figure 21) represents the process elements which are being performed such as activities which can either be atomic activities or sub-processes which are recursively refined by other activities.

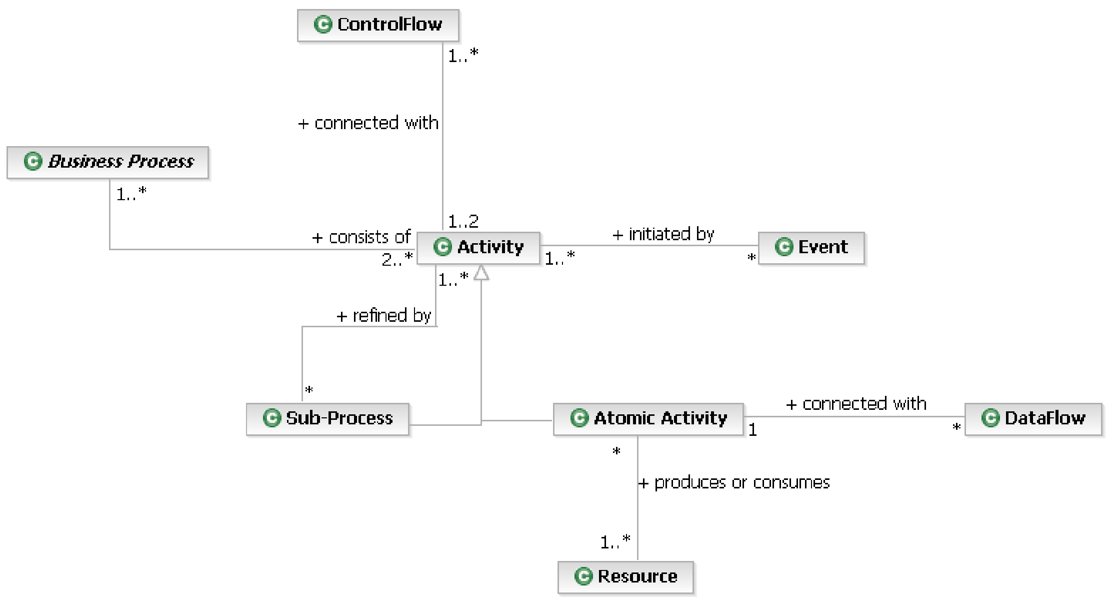

Figure 21: List/Korherr: Functional Perspective

According to the WfMC the organisational perspective represents where and by whom process elements are performed. This can be an organisational unit, a role, an (individual) human or an (automatic) resource. These can be divided into internal (belonging to the organisation) and external process participants. The automatic resource can itself be a software and either be an application or a service.

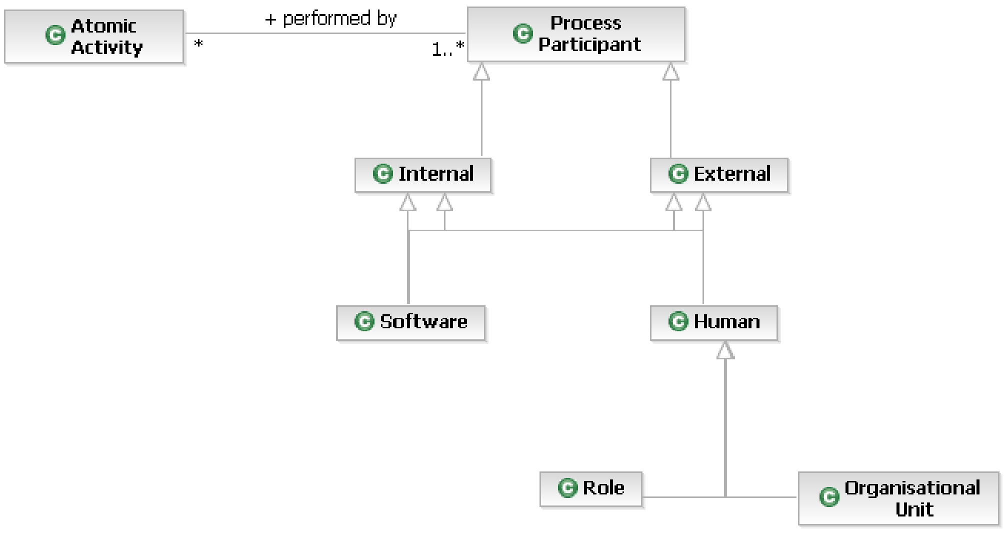

Figure 22: List/Korherr: Organisational Perspective

The behavioural perspective represents the sequencing of process steps as well as aspects how these processes are performed, e.g. through loops, iteration, decision process, etc. Besides simple control patterns (AND Split, XOR Join, etc.) there are advanced branching and synchronisation patterns (like OR Split, N-out-of-M Join) which according to the authors make sense for business process modeling.

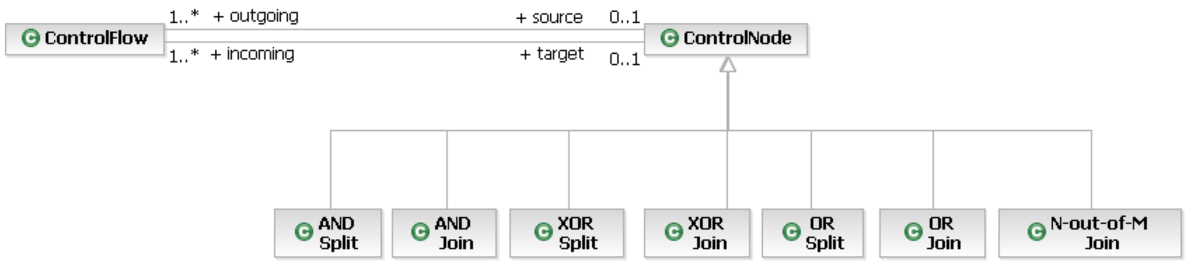

Figure 23: List/Korherr: Behavioural Perspective

Last but not least they describe the informational perspective which represents informational entities produced or manipulated by a process (data, artifacts, products and objects). This is inspired by the workflow data patterns as well as by the input/output view of ARIS.

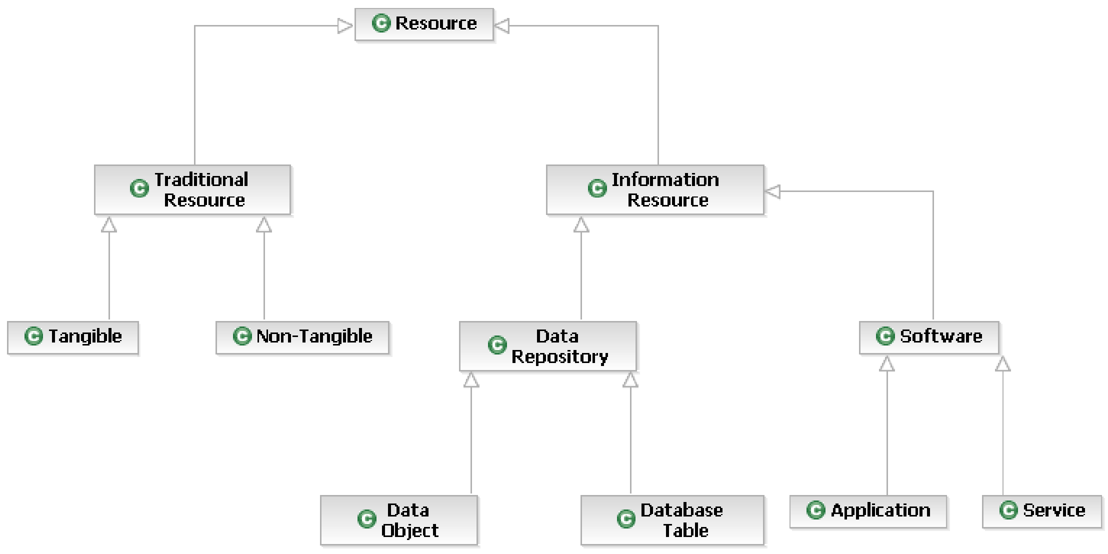

Figure 24: List/Korherr: Informational Perspective

3 Comparison

The following table shows all relevant aspects of the approaches mentioned in section 2 and compares them. The meta-model which covers most relevant aspects is BPDM, but it is also the most complex one. It is unrealistic to include all elements of BPDM in one diagram and henceforth in one model, but in reality most of these elements will be displayed in different diagrams. But in JWT we envision an easy-to-use and easy-to-understand modeling tool for the usage of different representations and different workflow engines. Some necessary elements like several events have been identified and will be included in an upcoming version of the tool set. Additionally we will investigate what constructs are missing for generating XPDL or WS-BPEL code (like e.g. Artifacts).

Concept	***Simple BPM***	***BPDM***	***AgilPro/JWT***	***EPC***	***List / Korherr***	***UML2 Activity Diagram***
General concepts						
Process	Process	Process	Activity	Function	Atomic Activity	(not exactly specified)
Process behavior	(no distinction)	Activity	(no distinction)	(no distinction)		Activity
Link to another process	SubProcessActivity	Sub-Process Activity	ActivityLinkNode	-	Sub-Process	CallBehaviorAction
Included Process	-	Embedded Process (subClassOf) Process Step Group	StructuredActivityNode	ComplexFunction		StructuredActivityNode
Group	-	Part Group	Group	-	-	ActivityGroup , ActivityPartition
Activity	Activity	Simple Activity	Action	ElementaryFunction	Activity	Action
Transition	Transition	Succession	ActivityEdge	Flow Connector	DataFlow, ControlFlow	ActivityEdge (ControlFlow, ObjectFlow)
Guard on Transition	-	Change	Guard / GuardSpecification	-	-	ValueSpecification
Loops	-	Activity / Conditional / Multi Instance Loop	-	-	-	LoopNode
Control nodes						
Process start	-	Start	InitialNode	-	-	InitialNode
Process finish	-	Finish / Terminate Event	FinalNode	-	-	ActivityFinalNode
Process flow abort	-	Abort / Error / Cancel Activity / Terminate Activity / Error Activity	(no distinction)	-	-	FlowFinalNode
XOR-Split	Route	Exclusive Split	DecisionNode	XOR	XOR Split	DecisionNode
XOR-Join	Route	Exclusive Join	MergeNode	XOR	XOR Join	MergeNode
AND-Split	?	Parallel Split	ForkNode	AND	AND Split	ForkNode
AND-Join	?	Parallel Join	JoinNode	AND	AND Join	JoinNode
OR-Split	-	Inclusive Split	-	OR	OR Split	ForkNode with ValueSpecification
OR-Join	-	Inclusive Join	-	OR	OR Join	JoinNode with ValueSpecification
IOPE						
Input data	Data	Interaction Flow	Data	Deliverable	Resource	InputPin
Output data	Data	Interaction Flow	Data	Deliverable	Resource	OutputPin
Precondition	-		-	-	-	Constraint
Effect	-		-	-	-	Constraint

Events						
Event	-	Behavioral Change / Interaction Flow / Happening Conditions	Event	Events	Event	Event
Message Event	-	Message	MessageEvent	-	-	MessageEvent
Timer Event	-	Time Condition on Start/Succession	TimerEvent	-	-	TimerEvent
Rule Event	-	Statement Change Condition on Start / Succession	-	-	-	ChangeEvent
Link Event	-	-	-	-	-	LinkAction
Multiple Event	-	-	-	-	-	-
Compensate Event	-	Compensation Connection	-	-	-	-
Error Event	-	Error Activity	-	-	-	RaiseExceptionAction
Business specific						
References	-	-	Reference	-	-	-
Business Function	-	-	Function	Information Object	-	-
Role	-	Processor Role / Performer Role / Actor / Pool / Lane	Role	Organisation Role	Process Participant, Internal, External, Role, Software	Swimlane
Organisation	-	-	OrganisationUnit	Organisation Unit	Organisation Unit	-
Application	-	-	Application	-	Application, Service	-
Parameter	-	-	Parameter	-	-	Parameter
Interactions						
Interaction	-	Interaction	-	-	-	(not in AD, but in Sequence Diagrams)
Message channel	-	Interaction Flow	-	-	-	-
Interaction role	-	Interaction Role	-	-	-	(not in AD, but in Sequence Diagrams)
Flow Binding	-	Flow Binding	-	-	-	-
Goals, Measures						
Goals	-	-	-	-	Process Goals, Enterprise Goals	-
Measure	-	-	-	-	Quantitative, Qualitative Measure	-
Deliverable	-	-	-	-	Service, Product	-

References

1. BPM-Guide: Bartonitz, M. *Wachsen die BPM- und Workflow-Lager zusammen?* Available online at http://www.bpm-guide.de/pic/xlarge/339.gif
2. BPM-Guide: Gille, M. Die nächsten drei Buchstaben – BPM. Available online at http://www.bpm-guide.de/pic/xlarge/291.gif
3. Object Management Group (OMG): *Business Process Definition Metamodel*, Final Submission, December 2006, available online at www.omg.org/docs/bmi/06-11-03
4. Object Management Group (OMG): *Unified Modeling Language (UML) Specification: Superstructure, Version 2.0, Final Adopted Specification*, July 2005, available online at http://www.omg.org/docs/formal/05-07-04.pdf
5. Object Management Group (OMG): *Unified Modeling Language (UML) Specification: Infrastructure, Version 2.0, Final Adopted Specification*, July 2005, available online at http://www.omg.org/docs/formal/05-07-05.pdf
6. Korherr, B. and List, B. *A UML2 Profile for Event Driven Process Chains* Proceedings of the 1st IFIP Interational Conference on Research and Practical Issues of Enterprise Information Systems, Vienna, Austria, 2006
7. List, B. and Korherr, B. *An Evaluation of Conceptual Business Process Modeling Languages*. In Proceedings of the ACM SAC'06, Dijon, France, 2006.
8. Curtis, B., Kellner, M. and Over, J. *Process Modeling*. Communication of the ACM, Vol. 35, No 9, 1992.