



BPMO Tutorial

Defining a Private Business Process in a Knowledge Base

By Dieter E. Jenz, President, Jenz & Partner GmbH

First Edition September, 2003 - Draft

Jenz & Partner GmbH
Hainstr. 40a
63526 Erlensee
Germany
Phone : ++49-(0)6183-9100-0
Email: info@jenzundpartner.de
Internet: <http://www.jenzundpartner.de>

Jenz & Partner was founded in 1985. Dieter E. Jenz serves as the company's president. The company provides a range of industry analyst, business and technical consulting, and educational services. It is widely known for its contributions to distributed applications, object-oriented development, and relational database technology. For additional information, Jenz & Partners' Website URL is www.jenzundpartner.de.

Jenz & Partner has developed a business process ontology, which can be made available to clients in the context of consulting engagements. Jenz & Partner provides training in ontology definition and in software development process optimization in general.

This White Paper focuses on human-oriented business process (workflow) aspects, which primarily execute within the organization boundaries, and are thus usually referred to as private business processes. It leaves so-called public business processes aside, which support the collaboration of business partners across enterprise boundaries.

The Protégé-2000 ontology and knowledge base editor has been used for the definition of the Jenz & Partner business process ontology. Protégé-2000 was developed by Stanford Medical Informatics at the Stanford University School of Medicine with support from various agencies.

Although this information is believed to be accurate at the time of publication, Jenz & Partner GmbH cannot and does not warrant the accuracy, completeness or suitability of this information or that the information is correct. Jenz & Partner GmbH and/or Jenz & Partner GmbH analysts and consultants cannot be held liable for any damages caused or alleged to be caused directly or indirectly by decisions made using any Jenz & Partner GmbH material. This publication and features described herein are subject to change without notice.

Names appearing in this document that are registered trademarks are not mentioned as being so, nor is the trademark symbol inserted with each mention of these registered trademarks. Product names and company names mentioned in this publication are the property of their respective owners. In no way does Jenz & Partner GmbH have the intention of infringing on any registered trademark.

Copyright © 2003 by Jenz & Partner GmbH. All rights reserved. No part of this publication may be reproduced by any method whatsoever, without the prior written consent of Jenz & Partner GmbH.

Table of Contents

Goals	1
Case Study: Private Mortgage Loan Application Processing	3
Defining the Business Process.....	5
Defining Business Entities	5
Defining Process Task Context Types	6
Defining the Business Process.....	10

Goals

The primary goal of the Business Process Management Ontology (BPMO) is to provide a stable platform for the semantically rich definition of business processes. The BPMO is based on open standards, some of which are still in development, but are considered sufficiently stable to base serious work on.

(Standards) Body	Description	Current Status
UN/CEFACT	UN/CEFACT Modeling Methodology (UMM)	N090R10, Nov. 2001, Draft Status
UN/CEFACT	ebXML BPSS (Business Collaborations)	Version 1.05, July 2002, Draft
UN/CEFACT	Core Components Specification (CCTS)	Version 2.0, August 2003
OASIS	Universal Business Language (UBL)	Version 0.8, September 2003, (based on CCTS)
BPMI.org	Business Process Modeling Notation (BPMN), defines a mapping to the Business Process Execution Language (BPEL4WS)	Version 1.0, August 2003, Working Draft
DAML.org	DAML-based Web Service Ontology (DAML-S)	Version 0.9 Beta, July 2003

The Business Process Modeling Notation (BPMN) is expected to gain broad acceptance in future business process modeling. BPMN has been developed under the auspices of the Business Process Management Initiative (BPMI.org). BPMN is a rich notation which simplifies the modeling of complex business processes. Many vendors of business process modeling tools have contributed in the development of the specification, with IDS Scheer being a notable exception. BPMN seems to have broad industry support, which manifests itself in the growing number of process modeling tools that support BPMN. Compared to other non-proprietary process definition languages, such as the Workflow Management Coalition's XML Process Definition Language (XPDL), BPMN is much more expressive and, in addition, provides a mapping to BPEL4WS, a specification currently under development within OASIS.

BPMN does not currently specify an XML language layer above BPM execution languages (currently only BPEL4WS). In addition, BPMN does not contain a standard mechanism for the exchange of business process diagrams between conformant tools.

Given that the current BPMN specification is just a step in the evolution, rather than the end of the evolution, it represents a considerable step forward in that it unites the tool-vendor community behind a single specification.

Despite these limitations, the BPMN specification defines semantics and is highly generic, so that it can map to multiple lower-level specification languages. BPEL4WS is just one of these lower-level specification languages, and the only one referenced in the BPMN specification. The BPMN specification rightly states that "process language specifications is a volatile area of work, with many new offerings and mergings"¹.

¹ BPMN Working Draft, p. 23

In a sense, the Business Process Management Ontology (BPMO) solves the two issues identified above. It provides a language layer above BPM execution languages in the form of an ontology layer, and, although indirectly, provides platform- and vendor-neutrality. Tool-specific representations of business processes can be generated from a single source: the BPMO. In the simplest case, an XSLT script would do the job in that it takes an OWL representation of the ontology as input and produces tool-specific output.

BPMI.org notes volatility in the process language specification space. In a similar fashion, there is also significant volatility in the tool vendor space. Hence, the ideal approach for a user organization would be to achieve process language specification independence, tool independence, and vendor independence. The BPMO is a step in this direction.

Case Study: Private Mortgage Loan Application Processing

In the context of the case study, we are going to define a business process, which is based on a real process in the mortgage industry. However, the process has been simplified for the sake of this case study.

We define a so-called “private” business process. While that used to mean an enterprise-internal business process up until recently, with the advent of Web Services, that notion has already changed. Since Web Services technology allows a private process to cross enterprise boundaries (in that a Web Service may be provided by a trading partner, for example), private processes are no longer confined to intra-enterprise processes. A better term would probably be “unilateral process”, since it does not require a business analyst to reach agreement with a business partner regarding the definition of a business process. A “multilateral process” would then be a business process that requires two or more business partners to reach agreement over the definition of a business process. ebXML business collaborations are a popular example of “multilateral” processes.

The business process that we are going to define governs the processing of Mortgage Loan Application forms received from an applicant over the Internet. For example, a prospective home owner wants to take out a mortgage. The mortgage bank offers an electronic Mortgage Loan Application form, which the applicant can fill out on-line at home. Once the applicant has provided information necessary for prequalification, the applicant receives an on-line offer, which includes all necessary details (e.g. interest rate, mortgage term, and so on). When the applicant has completed and submitted the form over the Internet, it is checked for consistency. If this check fails, the applicant is requested to provide the necessary information and re-submit the application form. Otherwise, a temporary loan reservation is made, which automatically expires if the applicant has failed to send the physical documents in time (within 3 working days). Supporting physical documents, such as earnings record, etc., are required for further processing of the mortgage loan application.

Once the necessary physical documents have been received through ordinary mail, a number of tasks are performed to determine whether the mortgage loan application should be accepted or rejected. The final decision is largely based on the outcome of two scoring steps. The first score, the credit score, reflects the credit-worthiness of the mortgage loan applicant, and the second score, the subject score, represents an assessment of the subject of the mortgage loan (e.g. a house, a building lot, etc.).

Each scoring result is interpreted based on pre-defined bandwidths, which may be likened to traffic lights. The highest bandwidth represents “green”, which means that the rating is sufficient for automated approval. The middle bandwidth represents “yellow”, meaning that a mortgage clerk has to review the case, make an informed decision and alter the score to either “green” or “red”. Hence, the outcome of the decision may be either approval or rejection. Finally, the lowest bandwidth represents “red”, indicating that the rating must result in automated rejection.

If the application has to be turned down due to bad scoring results, the applicant is sent a letter and the temporary mortgage loan reservation is canceled. If the application is accepted, mortgage loan processing continues and the mortgage loan applicant is sent a commitment letter with an attached credit agreement. In reality, this process is more complex. However, for the sake of this case study, we focus on a simplified version of the business process.

Another process would start when the credit agreement signed by the mortgage applicant is received.

We use the Protégé Ontology and Knowledge Base Editor tool, which has been developed by Stanford University as open source software. Currently, Protégé has a user base numbering more than 10,000 users. As a generic Ontology and Knowledge Base Editor, Protégé has not been designed with business process modeling and management in mind. However, Protégé includes a Graph Widget, which allows for the graphical representation of relationships between entities. As such, Protégé qualifies as a simple business process diagram editor. However, Protégé is no match to purpose-developed business process modeling tools. On the other hand, though, Protégé will still be adequate in many cases.

Protégé is an open source Java tool that provides an extensible architecture for the creation of customized knowledge-based applications. However, in this case study, we use Protégé as is, with no extensions.

Defining the Business Process

The BPMO is designed to support unilateral and multilateral (business collaborations a la ebXML) processes. Hence, it exceeds the current scope of the Business Process Modeling Notation (BPMN).

The BPMO is terminologically aligned with the BPMN. This is considered a major requirement, since there is currently a lot of confusion among business process modelers. Process modeling tool vendors use different terms for the same thing and identical terms for different things. For example, the term “activity” means very different things to different people. There is hope that the BPMN might put an end to that kind of confusion.

Defining Business Entities

When a process modeler starts out with the definition of a new business process, chances are that only few if any design artifacts exist that are necessary to fully describe the business process. For example, some business object definitions or business document definitions may not exist yet.

To facilitate business process modeling, the BPMO uses the concept of business entities. There are differing definitions of what a business entity means. The UN/CEFACT glossary defines a business entity as “something that is accessed, inspected, manipulated, produced, and so on in the business”. This is a very generic definition, and represents the highest level of abstraction. However, business people are used to talking in terms of business entities, and business entities are used in communication between business analysts and stakeholders at large.

Stakeholders and end users are not really interested how business entities are represented in software. For example, they are typically indifferent about whether a customer is represented as “party” in an object-oriented class model. By talking the language of stakeholders and users, business analysts avoid the trap of talking about things that interviewees fail to understand or understand differently. Situations where people talk about cross-purposes may often go unnoticed for some time.

Business entity and business object are not the same thing. For example, the business entities “customer” and “supplier” may be represented by a business object named “party”, which is a generalization of “customer” and “supplier”. On the other hand, multiple business objects may constitute a business entity. An ontology is perfect for the definition of relationships between the business entity and business object concepts.

The BPMO uses the concept of a business entity as a high-level concept. It serves as a glossary of terms that are relevant in the business. As a first step, a business analyst would identify business entities (e.g. customer information, mortgage, etc.). Software engineers may later associate business objects with business entities (there is a many-to-many relationship between business entities and business objects). By focusing on business entities, the business analyst is relieved from having to identify business objects at a very early stage in process modeling. In a sense, business entities may be viewed as business object candidates, although a single business entity may translate into multiple related business objects.

There is no mention of business entities in the BPMN specification. The specification more or less focuses on business process diagram objects, such as events, activities, and gateways. The BPMO requires the business analyst to define business entities as a prerequisite for the definition of process context task types.

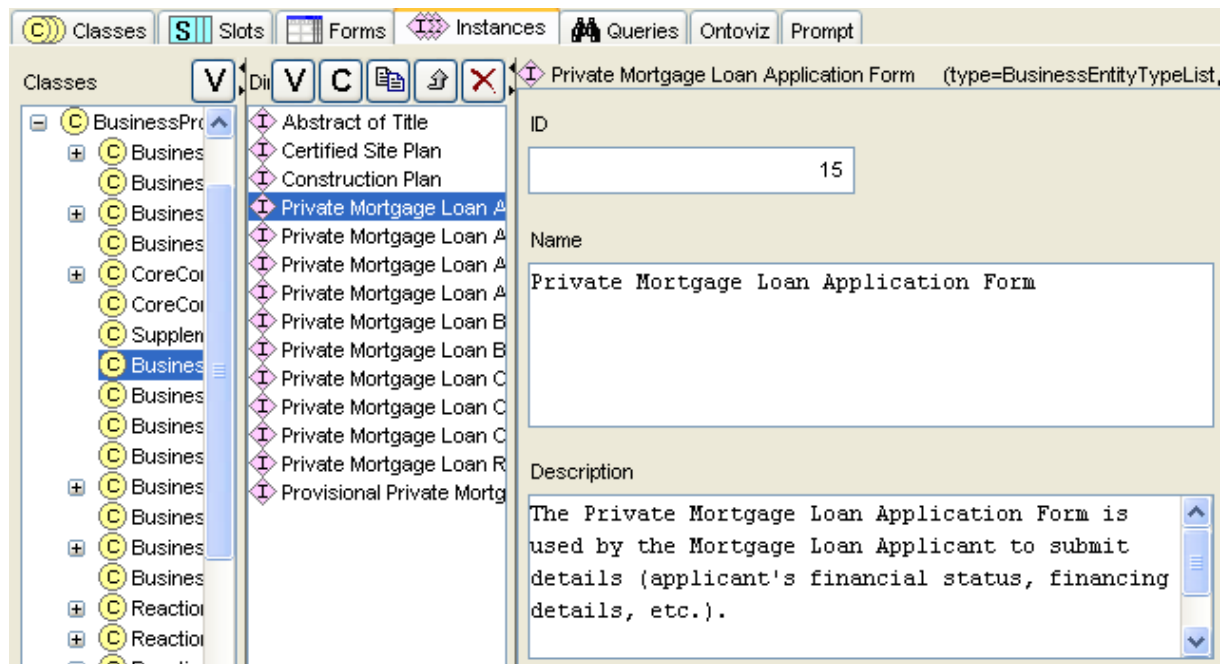


Figure 1: Business Entity Definition

The business analyst can add business entities as they are identified. That way, the business entity list grows over time. However, it is worth the effort to identify business entities as early as possible. This also helps to identify synonyms and homonyms, which is important, since, for example, it is seen very often that people working for different departments use different terminology to mean the same thing.

The business entity concept encompasses an abstract class with multiple subclasses. The business analyst would identify business entity subtypes as early as possible. Later on, when more details become available, the business analyst can describe further characteristics. This concept relieves the business analyst from having to describe business entities in detail, when such detail is not yet available.

Defining Process Task Context Types

The definition of Process Task Context Types requires an initial set of business entities to exist. This approach is based on the concept of reuse, allowing a business analyst to reuse a certain task context in multiple business processes. A well-defined business process ontology enables the business analyst to describe tasks with their full context, which is reusable.

A Process Task Context Type describes what role performs a task, the business entities and business documents it is related with, and the resources it consumes. It can be thought of as a logical building block. The Process Task Context Type concept proves very helpful when it comes to the generation of software artifacts. A generator tool can be used to create code from Process Task Context Type definitions.

A task is performed by a role, which is resolved at run time to determine the person(s) that are eligible to perform the task. A task may create, transform or consume information. It may also trigger the exchange of business documents to other tasks. A task is a logical entity. It is associated with a service (i.e. an application), which implements the task context. Technically speaking, a task can be implemented by multiple services. For example, there may be different implementations in multiple programming languages.

A task has the following properties:

- A task is performed by a single role in one location;
- A task can be performed within a fairly short period of time (seconds or minutes);
- A task represents a logical unit of work (i.e. a transaction)
- A task may be of one of the following types:
 - Manual task: performed by a role without involvement of an IT system;
 - Semi-automated task: performed by a role with involvement of an IT system. The role interacts with the system through one or more forms;
 - Automated task: performed by an IT system without involvement of a user role.

Every task represents a defined context, which includes the following items:

- Role: A logical abstraction of one or more physical actors, usually in terms of common responsibility or position. An actor may be a member of one or more roles. Example: Mortgage Clerk;
- Business Document: The set of information components that are interchanged as part of a task. A business document may participate in a message flow. Example: Private Mortgage Loan Application Form;
- Durable Information Entity: An information entity that a task needs to perform its function, which must be represented in a persistent storage mechanism, and whose state must exist beyond the lifetime of the service (application) that implements the task. It may be composed of multiple business objects. Example: Private Mortgage Loan Application Information;
- Resource: A real object that can be identified. Example: Flatbed Scanner.

Defining task context types is not as easy as it seems in the first instance. Process modeling is a top-down approach and may happen before Durable Information Entities and Business Documents are fully defined.

A business document is defined as the set of information components that are interchanged as part of a business activity or task. Sending and receiving tasks may run on different systems, which may be in different locations (e.g. on a buyer's and a seller's systems). Business documents (e.g. "Order", "Invoice") reference business entities. For example, when a business expert talks about "invoice", she/he usually means either a paper invoice form or an electronic representation of an invoice (e.g. an EDI representation). Both are business documents. At this time, a business expert does not care that there will be another representation of a business entity in the form of one or more business objects

A durable information entity represents a business entity in the end users' language, while a business object represents the software engineer's view. During business object modeling, a refactoring of business object classes (concepts) may become necessary. As a simple way out, it may seem best to associate durable information entities with task context types, rather than business objects. As noted above, a durable information entity may represent multiple object types. For example, the durable information entity "Party Information" may represent all "Party" business object classes.

Sadly, there is no silver bullet solution. If we associate durable information entities with task context types, we lose precision of meaning. Although reviewers will be happy with that level of precision, since task context type models are concise, there is a downside regarding software development: It will be much more difficult to generate code from a model. On the other hand, though, task context type models will gain more stability and will be less subject to change. In essence, by associating durable information entities with task context types, we pay tribute to the fact that, at the time of process modeling, we only know about business object candidates.

If we associate task context types with business objects, we gain precision of meaning. Reviewers of task context type models might be overwhelmed by a potentially large number of business objects. For example, reviewers from the stakeholder community at large are not interested in knowing that we have a “PartyRole” class, a “PartyNationality” class, a “ContactPoint” class, and so on. In addition, if business objects change at some later time, this invariably renders affected task context types invalid. Clearly, we want to avoid this situation.

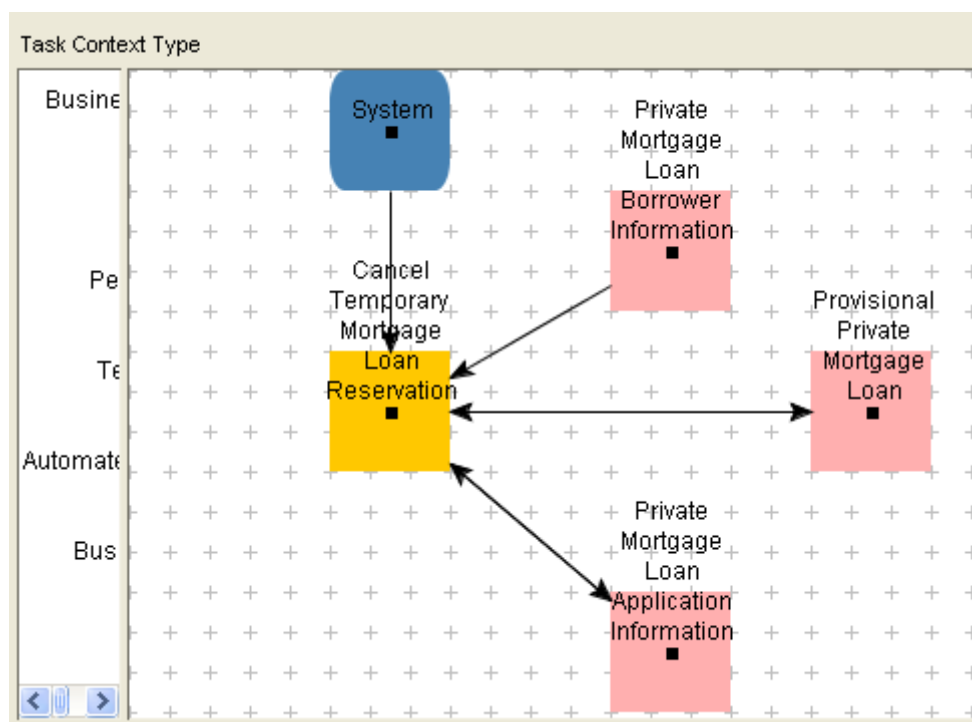


Figure 2: Process Task Context Type description for the “Cancel Temporary Mortgage Loan Reservation” task type.

As a pragmatic solution, task context types are associated with durable information entities. This effectively frees the business analyst from having to have a detailed knowledge of the business object level, which may or may not be very unstable at the time of process modeling. Like stakeholders, business analysts are not very interested in technical detail.

There is another concept, the “TaskInteraction” concept, that allows a business analyst to define user-task-interactions in more detail, and which is a better suited interface to software development. In this case study, however, we do not delve into that concept.

Task context types do not change frequently over time. It is more likely that new task context types are added. Once a few Task Context Types have been defined, the business analyst can reuse them in multiple business process definitions. A task is generally associated with a particular process definition, which prevents reusability. In contrast, Task Context Types may be associated with multiple tasks, thus providing them with context essential for the generation of software artifacts.

Process Task Context Types is a concept not rooted in BPMN. However, such concept has proven helpful in facilitating the generation of software artifacts. There is no conflict with BPMN, since the specification only focuses on business process definition and does not care about software artifact generation.

The specific value of the Process Task Context Type concept is its role as a bridge between business process modeling and software development. The semi-automated task context type definition below not only visualizes relationships between the task and durable information entities, but also includes other information:

- It links the task to an implementation, which may be a Web Service, a Java application, etc.;
- It identifies the initial form that the application will present to the user when the application is invoked. Alternatively, the application may be instructed to automatically generate a form at execution time;
- It associates the task with simulation information to be interpreted by a process execution simulation tool.

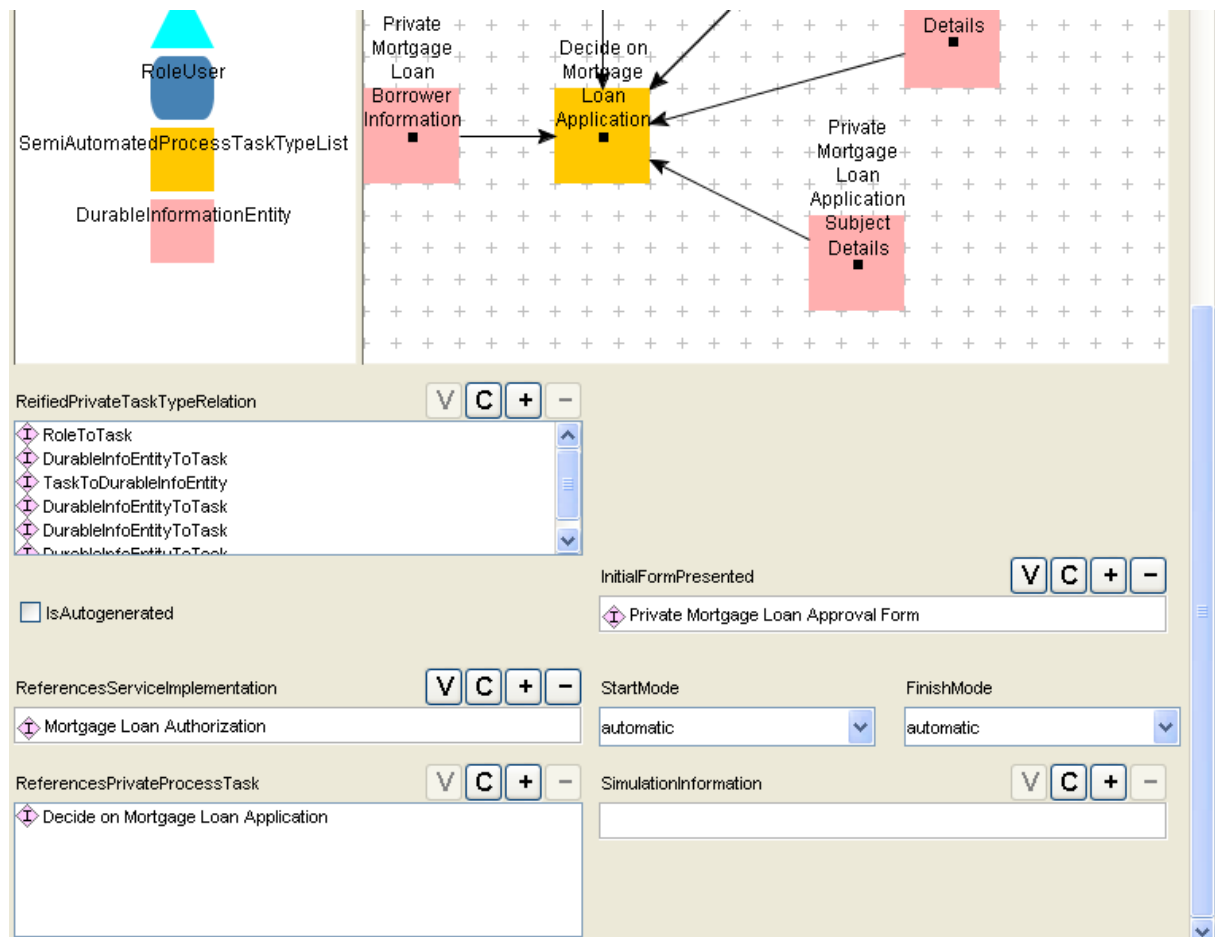


Figure 3: Detail Process Task Context Type definition

Process Task Context Type information will be used by generator tools, which can use this information to generate code or other software artifacts. Obviously, a generator tool would need to inspect both the process task information and the process task context type information in order to determine what exactly to generate. Process task information may override process task context type information.

Defining the Business Process

In preparation for the creation of a graphical representation of a business process, the business analyst creates definitions of process model artifacts, such as process tasks, sub-processes, and so on. Every individual task is assigned to a single business process. Hence, an individual task definition cannot be shared among multiple business processes. However, each task is associated with a process task context type, which is reusable.

The business analyst would not need to define the complete set of nodes (artifacts) before starting with graphical business process modeling. Process definition is a highly iterative process in its own right, and the BPMO supports this approach.

A business analyst instantiates the “ProcessFlow” concept to define a process flow, which is assigned to exactly one business process definition. The form contains a Graph Widget, which allows the business analyst to graphically design the process flow.

The Graph Widget allows a business analyst to define connectable nodes (i.e. events, tasks, gateways (connectors) on the fly when needed, or use existing definitions. The business analyst usually will prefer using existing definitions.

In the current version, the Graph Widget does not support user-defined shape libraries. Hence, the business analyst cannot use the BPMN shapes for process modeling in Protégé. User-defined shape libraries will be supported in the mid-term future, though. Since Protégé is not a process modeling tool, it is also not possible to draw process pools and “swim lanes”. Even with these restrictions, the functionality of the Graph Widget will prove sufficient in many cases.

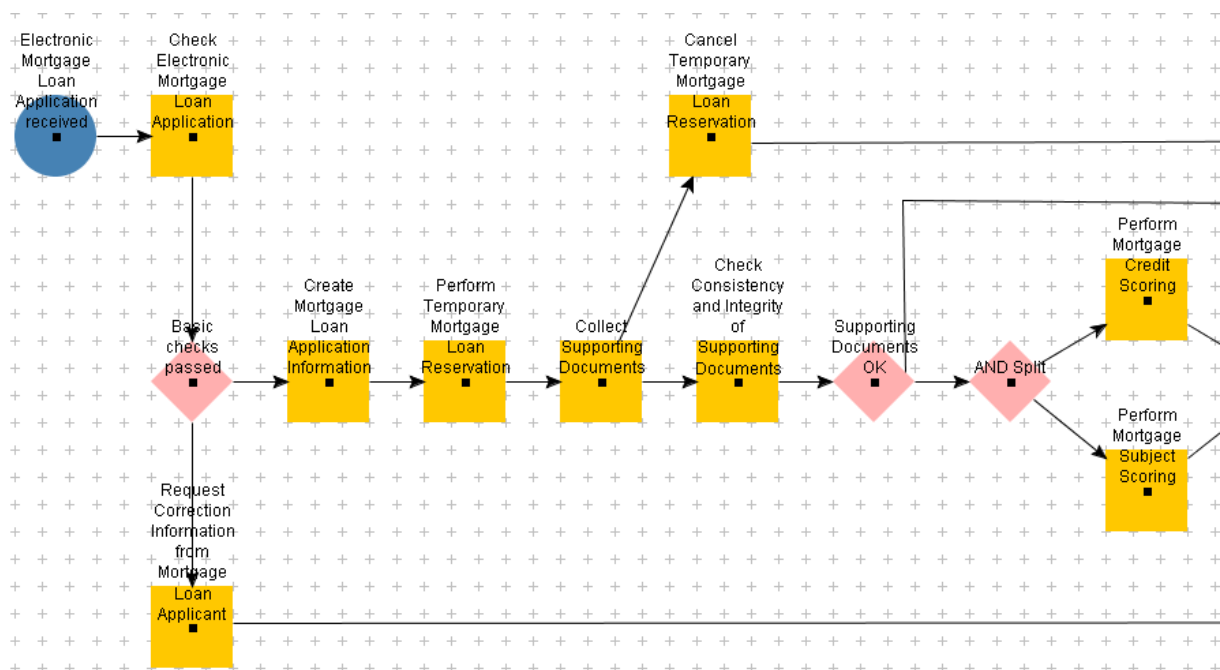


Figure 4: Private Process Definition (snippet)

The business analyst can edit task properties by double-clicking on a task shape. Protégé brings up a form, which presents information about that task to the user. The task window can serve as a stepping stone to display further information about some information item. For example, the user may wish to obtain further information about the process task context type the task is associated with. By double-clicking on the “PrivateProcessTaskContextType” field, the user can have Protégé display that information.

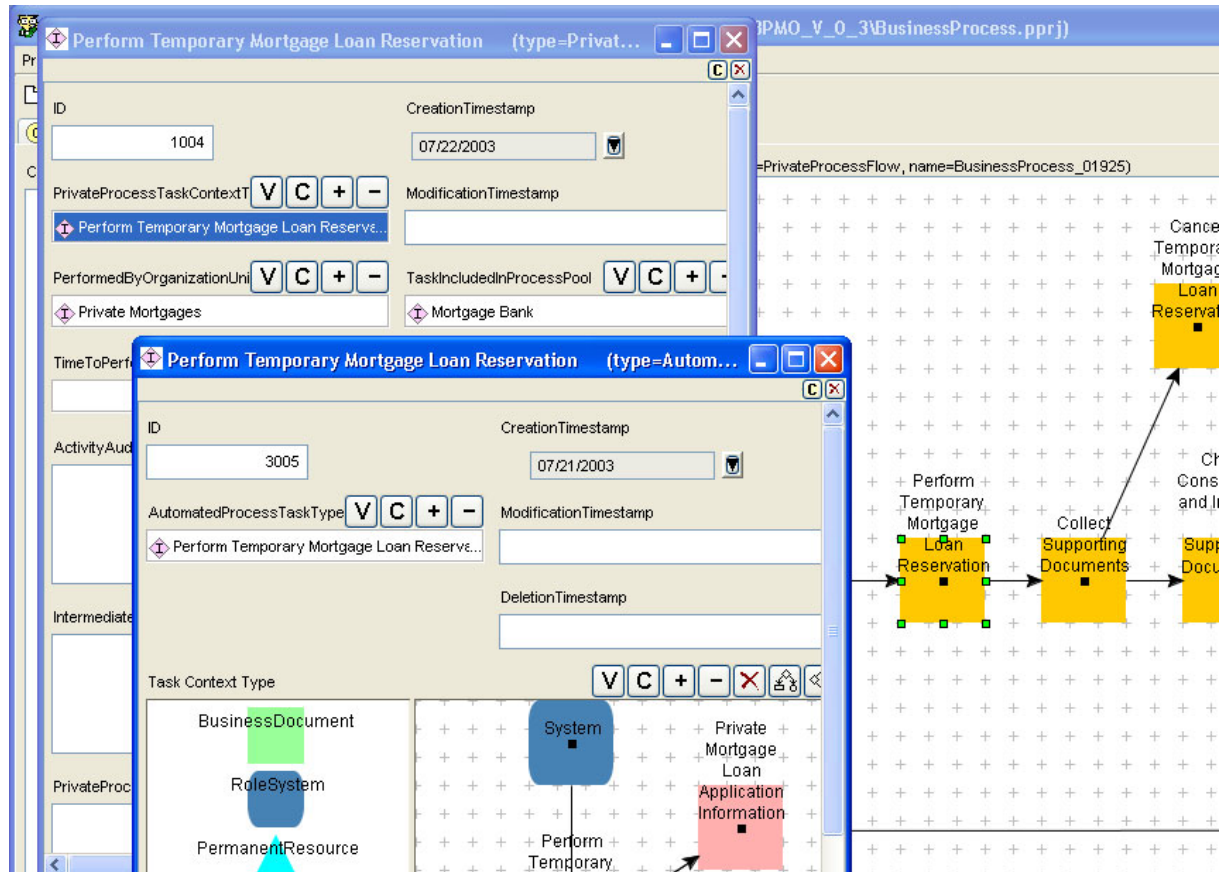


Figure 5: Displaying further information about a task (snippet)

In a similar fashion, the user can assign guard conditions to transitions outgoing from a gateway. In the following example, the overall score is determined. If the Mortgage Loan Applicant's Financial Status Score is "yellow" or if the Mortgage Subject Score is "yellow", it has to be explicitly decided whether or not to grant the mortgage loan. In all other cases, the decision can be reached without manual intervention. For example, if the Applicant's Financial Status Score is "red" and the Mortgage Subject Score is "yellow", the mortgage loan would not be granted.

The business analyst can define condition expressions and assign them to outgoing sequence flows. In addition, the business analyst may define an execution probability for each outgoing sequence flow. An execution probability figure is helpful in the simulation of a process instance execution.

To define a condition expression, the business analyst double-clicks on a sequence flow. A window opens which allows the user to enter data. By double-clicking on the "ConditionGuardPrivateProcess" field, the user can view the condition expression.

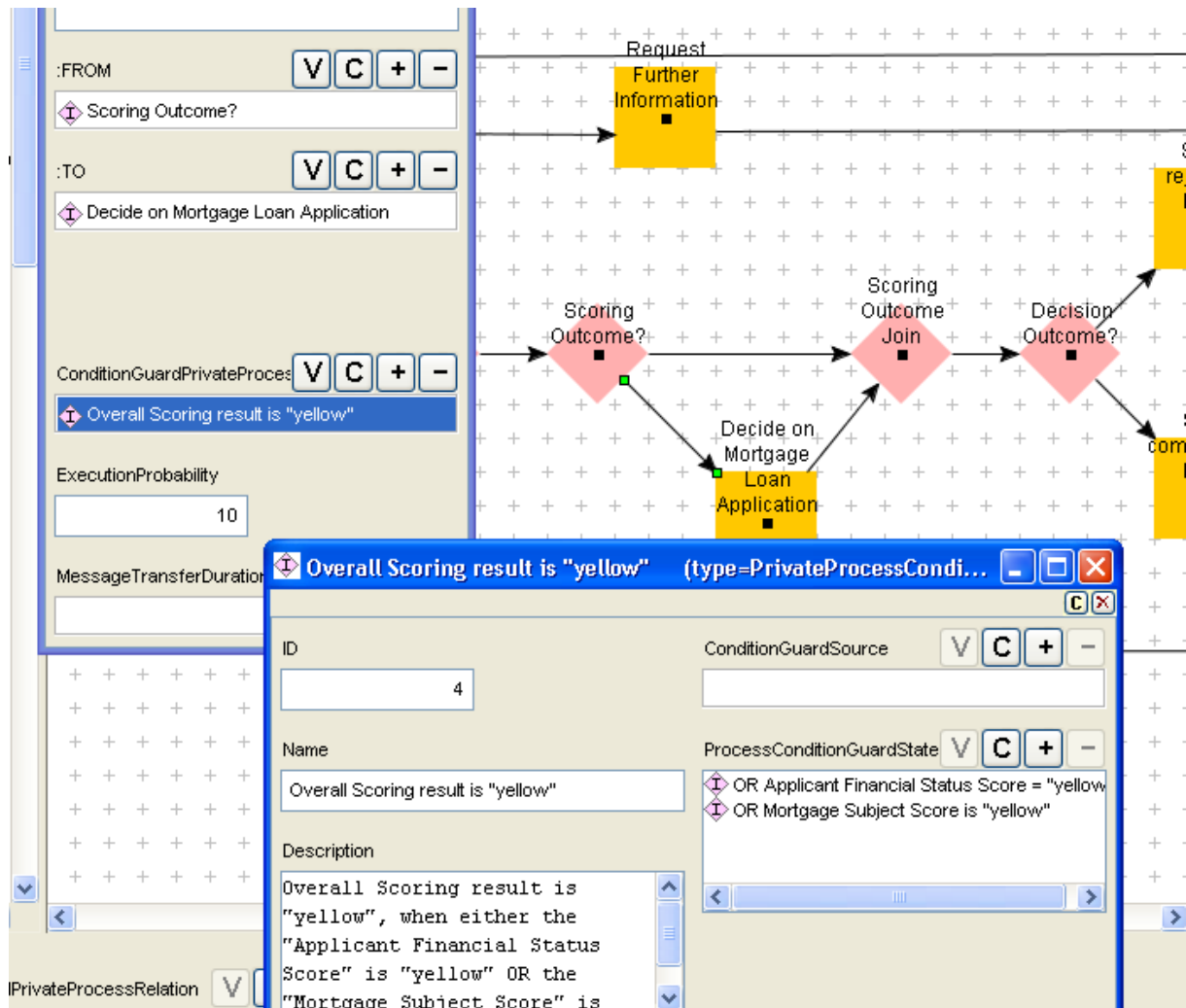


Figure 6: Assigning a guard condition to an outgoing sequence flow

Protégé features a simple knowledge base editor. In the current version, the editor can only read stored instances from the knowledge base and present them to the user in forms on the screen, or store data in the knowledge base. There is no way for a business analyst to define business logic. In technical terms, Protégé implements a two-tier model, consisting of the presentation layer and the data storage layer. For example, it would prove helpful to define queries that select only those tasks that are still available for assignment to a business process. Probably, future versions of Protégé or generic OWL editor tools will add such functionality.

BPMN supports graph-structured diagrams rather than block-structures like BPEL4WS. The greater flexibility comes at a cost, however. A business process analyst can create process definitions that are not executable or will behave in a manner neither intended nor expected by the business analyst. For example, in complex process definitions, a business analyst may easily overlook design errors, such as improper looping or deadlock situations. It is therefore necessary that a business process modeling tool provides a “validate business process definition” function, which lets a business analyst perform basic sanity checks.

An ontology and knowledge base editor, such as Protégé, does not provide business process model validation functionality. This is not a weakness, since an ontology and knowledge base tool is not expected to support such feature. There are several solution paths:

- Someone develops a plug-in for an ontology and knowledge base tool, so that the business analyst can perform validation without having to leave the tool;
- Constraints can be defined about a knowledge base. The Protégé Axiom Language (PAL) could be used to express constraints;
- The business analyst imports a business process definition into a business process modelling tool that has a process model validation feature;

In all, the BPMO is proof that an ontology forms a suitable platform for the definition of semantically rich business process definitions.