Specifying Flexible Business Processes Using Pre and Post Conditions

Jeroen Grondelle, Menno Gülpers

Specifying Flexible Business Processes using Pre and Post Conditions

Jeroen van Grondelle, Menno Gülpers

Be Informed BV, Apeldoorn, The Netherlands
{j.vangrondelle,m.gulpers}@beinformed.com
http://www.beinformed.com/

Abstract. Today's business processes have to address many, complex requirements. Mass customization leads to personalized, contextual products being offered by governments and enterprises and, as a result, the business processes for selling and offering these products are divers and contextual as well. At the same time, regulations in the area of compliance and a growing rate of change introduce additional complexity. These developments pose major challenges to the field of business process modeling. Conventional process modeling, in terms of activities and the flow they are executed in, has proven to lead to complex and often rigid business processes.
In this paper, we present our experiences with specifying business processes based on activities and their pre and post conditions instead of flow. The resulting business processes are flexible: they allow knowledge workers to influence their own process and they do not require the explicit modeling of flows to deal with exceptions and switching between straight through and human processing.
Our formalism facilitates an agile modeling process. The formalism helps involving business users in modeling as it can be expressed well into natural language. Furthermore, it allows for separation of concerns in modeling by having an algorithm consolidate the different areas of requirements into an executable business process. Analysts can focus on modeling the different concerns and are no longer required to manually consolidate all the requirements into a business process that is believed to address all of them.

Key words: Business Processes, Adaptive Processes, Goal Orientation, Business Rules, Complexity, Natural Language Generation

1 Introduction

Today's organizations are dealing with a number of trends that increase the complexity of their business. Most governments and enterprises offer products that have many variants and options, depending on the customers' context and his individual choices. Enterprises use this as a marketing tool, increasing their revenue by addressing multiple target groups with specific products. Governments offer products, like grants, taxes and permits, that are the result of complex

policies. The business processes that are introduced to produce, sell or apply for these products are often equally individual and contextual of nature. They need to offer the knowledge workers that perform them the flexibility that matches their experience, while at the same time guaranteeing consistency and quality of the result.

At the same time, organizations need to deal with a growing rate of change. Enterprises want to react to changing market circumstances ever faster, resulting in changes to both their products and the supporting business processes. As a result of regulation, the rules they must comply with change or new constraints are introduced.

This agility typically poses requirements to the modeling process. Changes have to be implemented often and fast. This is specifically a challenge in the area of validation, as conventional testing of all possible scenarios is no longer feasible or at least takes too long. Reviewing by business experts and the ability to trace models to their source are becoming more important. Also, the typical IT approach of handing over a specification between disciplines like analysis, functional design and engineering consecutively has become a bottleneck. Equally important, agility introduces requirements of its own to the business process itself. When changes occur frequently, business processes will have to deal with active processes for old versions of products. Either by migrating transparently to the newest product definitions or by completing processes against the policy that was in place when the process started.

Classically, IT has defined business processes in terms of the activities that are performed in an organization and the order in which this is done. Conventional process modeling standards are available, like OMG's Business Process Modeling Notation [1] and the Business Process Execution Language [2]. More recently, the terms Dynamic Case Management [3] and Adaptive Case Management [4] were coined for more dynamic, rule oriented approaches, that address the fact that the complexity we described earlier has proven to pose a challenge when using the metaphor of flow. It typically results in processes with a lot of forks to accommodate process variants or exception flows. Alternatively, a large number of processes is often created that match the large number of product variations, even when these process variants essentially perform the same task.

Using declarative techniques in process modeling is broadly seen as a way to overcome limitations of conventional, imperative approaches, both in industry and academics. In for instance [5], Goedertier and Vanthienen describe the differences between declarative and imperative process modeling and how a declarative approach helps to model the actual business concerns instead of, with an imperative approach, model a process flow that meets these constraints implicitly. In [6], Pesic and Van der Aalst introduce a ConDec language that attempts to reduce over specification and introduce flexibility by using declarative techniques to specify business processes. In [7], Schonenberg et al. describe different aspects of flexibility required in today's business processes. Also in this paper, replacing flow by declarative (precedence) constraints is presented as a source of flexibility. Andersson et al. address flexibility, traceability and business orienta-

tion in [8] by introducing an activity dependency model that, like our formalism, focusses on the type of dependencies that exist between activities.

This paper introduces the formalism we use in Be Informed's Business Process Platform[1] to specify business processes in a declarative, goal oriented way. It leaves consolidation, and the combinatorics and order that follow from it, to be performed by an algorithm, so that enterprise modeling professionals can focus on modeling the underlying business aspects in terms of requirements. The resulting processes can be completely prescriptive, but at the same time offer great flexibility. They allow experts to influence their own work and they deal with exceptions well.

2 Specifying Flexible Goal Oriented Processes

In this section we introduce the formalism we use for specifying business processes.

2.1 Pre and Post Conditions

The formalism is based on the notion of activities, the pre conditions that have to be met for these activities to be performed and the consequences that result from these activities, expressed in terms of post conditions.

Be Informed uses a graph oriented representation consisting of concepts and relations between concepts. Concepts and relations have a type and can have properties. Multiple labels and fragments of text containing definitions, examples etc. can be associated with concepts, as can references to the underlying content that is the source of a concept or where it occurs. The types of concepts and relations that are available in a model are introduced in a meta model, that also contains type hierarchy rules and constraints about which relations and properties may occur at concepts of which type.

The central types in the meta model discussed here are the concept types that represent cases and the activities performed within cases. Furthermore, it contains the conditional relation types that capture pre conditions and an abstract relation type called Consequence which has post condition semantics. The meta model is summarized in Figure 1.

Nodes in Figure 1 introduce concept types, edges indicate that relations of the specified type may occur between the concept types it connects. For instance, the relations between Activity and Decision introduce both a pre and a post condition. The Requires Taken relation represents that instances of activity may require a specific decision to be taken before the activity may be performed. The Decides relation introduces the possibility for an activity to have the taking of a specific decision as post condition. This can be both read in a formal, post condition way and a more intuitive, procedural way. In the first case, the activity is only completed if the specified decision is taken. The informal way would be

[1] `http://www.beinformed.com/`

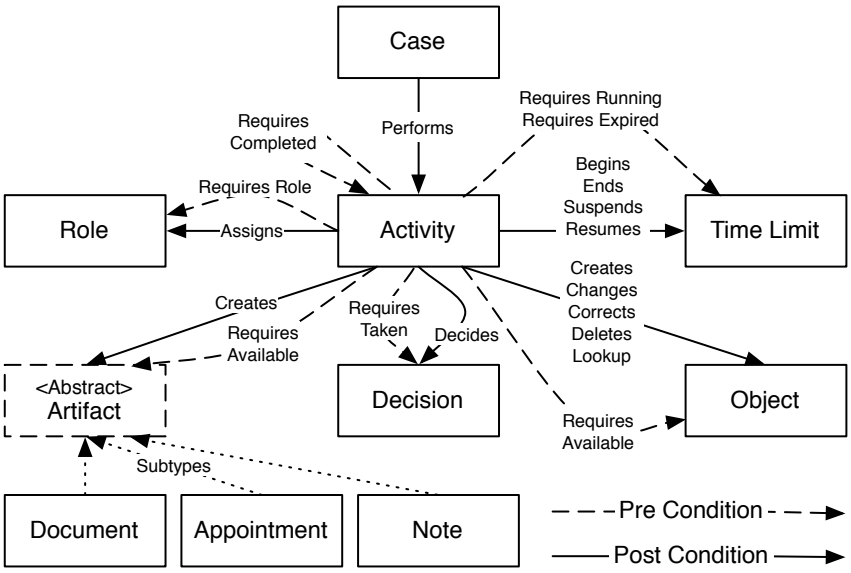

Fig. 1. Summary of the Meta Model for capturing Business Processes

to state that the decision is taken as part of the activity. The latter is correct in most cases, but overlooks the fact that the decision may have been taken in another context. The formal, correct interpretation allows the activity to be completed in such cases.

The meta model introduces **artifacts**, that represent documents produced, meetings held and notes taken during the process. Creation of artifacts is typically modeled as a consequence of an activity, their existence is often a pre condition for performing other activities. For instance, in Figure 2 an intake meeting is planned as part of the activity of registering the application form, the existence of an intake form is a pre condition for performing an assessment activity.

The meta model also enables the modeling of data **objects** used in a case. These objects can be local, case scoped objects or persistent objects in a registration. This difference is omitted for the remainder of the paper. Storing or updating objects is typically modeled as a consequence of an activity, their availability is typically a pre condition.

The meta model identifies the **decisions** taken as part of the process. Often implicitly embedded in process flows, we believe decisions are a core concept in today's business, as they capture a major part of what people do in organizations today. Is this customer interesting to us? Is this citizen entitled to a grant? And if so, how much is he entitled to? Within the business process meta model, decisions are identified and if necessary acted upon. The definition of the decision itself

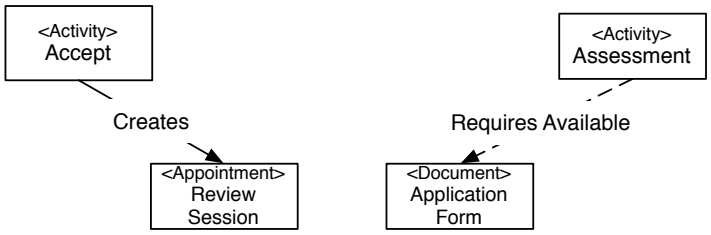

Fig. 2. Relations between Activities and Artefacts

is done in separate model fragments, with a different, decision oriented meta model.

In people centric processes, modeling the **people involved** is important. Our formalism has the notion of both user roles and involvement roles. User roles are typically used to represent user competences, responsibilities etc. Involvement roles encode similar aspects, but specific to the role a user plays in an individual case. Issuing/assigning these roles can be a consequence of an activity, having the appropriate roles is a typical pre condition.

An important aspect of a business process is the applicable **time limits**. In our formalism, activities may begin, end, suspend or resume a time limit. Other activities may require a certain time limit to be either still running or already expired as a pre condition. For instance, Figure 3 shows a maximum response time to a grant application that is specified as a time limit, which is suspended for the time the citizen uses to produce additional information on request. The retention period of a case starts on publication of the decision and the archiving activity requires this period to have expired before archival may be performed.

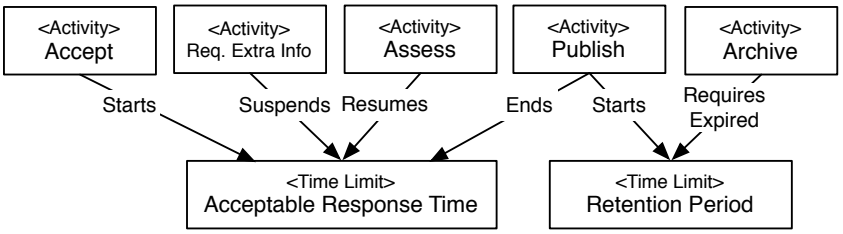

Fig. 3. Relations between Activities and Time Limits

2.2 An Example: Grant Applications

The model in Figure 4 captures the business process of applying for a grant. It introduces activities for accepting, assessing and archiving the application for the grant and publishing the decision. Assessing the grant application is done by deciding whether the grant is eligible, which can result in a confirmation or rejection letter. Time limits monitor the response time and the retention period.

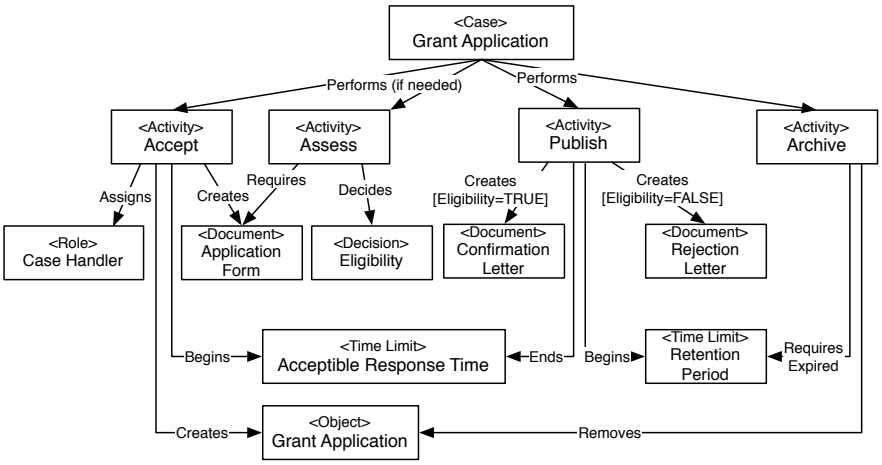

Fig. 4. A Model for Handling Grant Applications

The case Grant Application has to perform the activities Publish and Archive to be successfully completed and can perform the activities Accept and Asses. The latter two activities are not mandatory, their post conditions (and not the activities themselves) act as pre conditions for other activities. The creation of an application form is a pre condition for the assessment activity, in which a decision is taken regarding the grants eligibility. Publishing the decision requires that this decision has been taken, since the creation of a confirmation or rejection letter depends on the outcome of the eligibility test. However, the eligibility could be determined in another way, since the activity Asses itself is not mandatory, resulting in the Publish activity's pre conditions to be met. Accepting a grant application not only issues a case handler and creates an entry in the grant application registration, but also starts a time limit that monitors an acceptable response time for the application of the grant. Publishing the decision will end this time limit, while at the same time starting another time limit, defining a reasonable retention period, for instance 5 years. The archiving activity can only be performed when this period has ended and results in the removal of the grant application from the registration.

3 Goal Oriented Business Processes

Although the formalism does not explicitly describes flow, an executable process can be inferred from it. As its pre conditions are continuously evaluated, the inferred process is highly dynamic and responds well to user choices and external input.

3.1 Inferring the Process

Based on a business process described using this formalism, at any time, the activities that *may* be performed next can be determined based on case state by checking which activity's pre conditions are met. This information can be used to automate a process, by offering users only the tasks that may be performed. That way, all activities will be performed only when their constraints, in terms of order, availability of other information, competence of the actor etc., are met. When a model has many pre conditions between activities, effectively encoding order, this will lead to a conventional process, where each activity may be performed if its predecessor(s) are completed. In a more complex model, this strategy typically leads to many activities that can be performed, without being prescriptive which should be performed first.

Based on the same formalism, it is possible to infer which activities *need* to be performed in order to meeting an overall goal. In our formalism, the goal is modeled by expressing post conditions at the case level. Inferring the goal oriented process is done by (recursively) inferencing which activities contribute to meeting the pre conditions of activities that contribute to the goal. An activity can contribute to meeting the preconditions of another activity directly, if the activity is a pre condition itself, or indirectly, when post conditions of a particular activity are pre conditions of the other activity.

Post conditions can be conditional or optional. Conditional post conditions might encode activities or artifacts only required under certain conditions. For instance, a grant notification needs only be sent if eligibility for the grant is established. Optional post conditions are strictly not post conditions, as they may or may not be met. Typically, these are only met if required by some other activity's pre condition.

3.2 Flexibility through Goal Orientation

The fact that goal oriented processes focus more on the requirements that need to be met than specifying in which specific way to meet them, guarantees a large degree of flexibility.

Goal Orientation allows Knowledge Workers to Influence their Own Process Both the activities that may be performed and the activities that need to be performed can be presented to users. Performing the activities that need to be performed is usually the most straightforward approach for users, but experienced professionals may choose to perform activities that may be

performed, but are not inferred to be needed at this time. Reasons to do this might include availability of the required information for the activity (only) at this moment or the expert judgement that a case will for instance be accepted anyway, that way predicting that the activity will be inferred to be needed in the future.

Figure 5 shows a user interface that shows the process state to a user. It distinguishes between activities that have been completed, activities that need to be performed now, activities that are not needed but are allowed and activities that may not be performed at this time. Based on the type of pre condition violated, the user interface provides feedback on why an activity may not be performed. In cases where no activities are available to a user, this is crucial to allow him or her to assess how to progress the case: Is it my role that prevents me from performing the necessary activities? Is a lack of information blocking at this time? Is it time constraints that prevent this case from progressing?

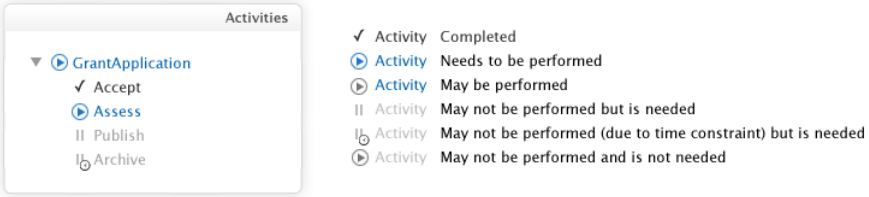

Fig. 5. Presenting Available Activities to a User

Goal Oriented Processes allow for (Ad Hoc) Interventions Another form of flexibility arises when flow oriented pre conditions are replaced by pre conditions on availability of data or information. Flow oriented pre conditions make it hard to override or repair a process when exceptions occur. If a certain activity is not performed, the activities that depend on it will never be performed. If, instead, the activity depends on the post conditions of the first activity, these pre conditions might be met in an alternative, possibly ad hoc, way, without the earlier activity ever being performed.

In Figure 6, the Application Form document required for performing an assessment is normally created in the Accept activity. However, providing the document directly in the document management system as part of an intervention would still allow the process to proceed as all pre conditions of assessment are met.

Flexible Switching between Manual and Straight Through Processing Many organizations execute their processes in a hybrid model: Part of the transactions are performed automatically and straight through, others require human intervention and are processed manually.

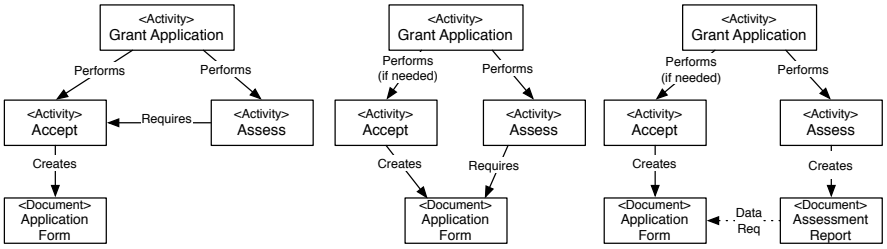

Fig. 6. Explicit and Implicit Dependencies between Activities

Typically there are two approaches used to distinguish between automatic and human work: Use explicit rules to evaluate which cases need to be processed by humans or attempt automatic processing by default and have users process the cases that terminate incompletely or lead to errors in the automated process.

In both cases, a major challenge is getting cases back into automated handling as soon as the need for human involvement is no longer present. Often, cases that fall out of the automated, straight through processing need to be processed manually from then on, even if large parts of the remaining work do not strictly require human intervention. In a flow oriented process, the only alternative is to explicitly bring cases back into the straight through processing flow, but this typically needs to be done individually for every possible reason a case might have left the straight through processing flow.

Goal oriented processes allow for far more transparent switching between manual and automated processing. For each case that terminates incompletely in straight through processing, it is known which pre or post condition failed. Instead of defining flow to deal with that, these failed conditions define classes of work in themselves, that can be assigned to human users. On resolving the pre condition, the case is available for straight through processing once again. Obviously, there might be additional issues to be solved by humans, but this approach guarantees that only tasks that really need it are processed by humans, while automated processing can always be attempted again transparently if the human task is completed.

Goal Related Feedback on Process Quality Traditionally, feedback on the performance of business processes is collected by reporting on counts and time related aspects across groups of process instances. It is no problem to derive production reports on activities performed, reports on adherence to time limits and distribution of possible outcomes from the processes inferred from our formalism.

The fact that the process is inferred from pre and post conditions introduces additional reporting possibilities that are closely related to the goals that are to be met in the business process. The same metadata that is used to present users with the activities available to them, as depicted in Figure 5, can be used to report on the reasons why processes weren't completed in an STP fashion

for instance. By reporting on which pre conditions were violated, very direct feedback is available that can be used for improving the business process. Is it the availability of information that prevents the process from being completed in a single transaction? Or are activities performed by users with insufficient expertise levels leading to reassigning of cases?

4 Verbalizing Process Specifications into Natural Language

Apart from being flexible, the business processes need to be up to date and reflect all the changes that organizations deal with on a regular basis. One of the ways to achieve this is to actively involve the business users and domain experts in the modeling of business processes. The main challenge in involving business users in enterprise modeling is the fact that most business users are not trained in formal modeling techniques. A formal, concise, visual representation can be quite intimidating to the uninitiated. One way of enabling business users to get involved in formal modeling is the use of natural language. Verbalizing graph oriented formalisms into (pseudo) natural language has been studied quite extensively, for instance by Funk et al. [11] and Kaljurand et al. [10], and we have good experiences with a similar approach based on pattern sentences [9]. Applying these techniques to the formalism introduced in Section 2 turns out to produce a very useful visualization of the business process models. Below is a summarized grammar of pattern sentences that match the presented meta model.

1. "A C case is only completed if"
   a) "the activity A is completed." $\leftrightarrow \{Case, performs, Activity\}$

2. "The activity A may only be performed if:"
   a) "the activity A' is completed" $\leftrightarrow \{Activity, requires, Activity\}$
   b) "a document of type D is available" $\leftrightarrow \{Activity, requires, Document\}$
   c) "the user is involved with role R" $\leftrightarrow \{Activity, requires, Role\}$
   d) "the decision D was taken earlier' $\leftrightarrow \{Activity, requires, Decision\}$
   e) "the time limit T has expired' $\leftrightarrow \{Activity, requiresExpired, TimeLimit\}$
   f) "the time limit T is still running' $\leftrightarrow \{Activity, requiresRunning, TimeLimit\}$

3. "The activity A is only completed if:"
   a) "a document of type D is created" $\leftrightarrow \{Activity, creates, Document\}$
   b) "the decision D is available" $\leftrightarrow \{Activity, decides, Decision\}$
   c) "a user was assigned role R" $\leftrightarrow \{Activity, assigns, Role\}$
   d) "time limit T has begun" $\leftrightarrow \{Activity, begins, TimeLimit\}$
   e) "time limit T has ended" $\leftrightarrow \{Activity, ends, TimeLimit\}$
   f) "objects of type O have been removed" $\leftrightarrow \{Activity, removes, Object\}$

For each pattern sentence, the left hand side represents the human readable representation, the right hand side triple represents which relations in the meta model it encodes. For instance, the subsentence "the activity A is completed" is used to encode any triples of the form $\{Case, performs, Activity\}$. The fact

that it is a pre condition according to the meta model is represented by prefixing it with the sentence part "A Case C is only completed if".

On verbalization, the pattern sentences are matched to the triples in the model, and the applicable parts are concatenated into complete sentences. Verbalization of a subset of the example in Section 2.2 using Be Informed Studio leads to the following sentences.

1. A Grant Request case is only completed if
   a) the activity Publish is completed,
   b) the activity Archive is completed

   and if needed
   a) the activity Accept is completed,
   b) the activity Assess is completed.
2. The activity Accept is only completed if
   a) a document of type Application Form is available
   b) an object of type Grant Request is available
   c) a user is involved with role Case Handler
   d) the time limit Acceptable Response Time has begun
3. The activity Assess may only be performed if
   a) a document of type Application Form is available
4. The activity Assess is only completed if
   a) a decision of type Eligibility is taken
5. The activity Publish is only completed if
   a) if $Eligibility = true$, a document of type Confirmation Letter is available,
   b) if $Eligibility = false$, a document of type Rejection Letter is available,
   c) the time limit Retention Period has begun,
   d) the time limit Acceptable Response Time has ended.
6. The activity Archive may only be performed if
   a) the time limit Retention Period has expired.
7. The activity Archive is only completed if
   a) objects of type Grant Application have been removed.

The pattern sentences used include feedback on the semantics of the meta model. The fact that pre conditions determine whether activities may be performed is a typical example where a modeling professional keeps in mind when interpreting the relation types used to encode pre conditions, while a business user needs to be reminded of this permanently. By including that explanation, it is automatically repeated for each activity and its pre conditions.

As we have shown in [9], pattern sentences have another important benefit: Every model can be verbalized using different pattern sentence grammars to support different expert levels and target groups. The same holds for verbalizing the models into multiple languages. Apart from translating the pattern

sentences, this requires the localization of the models, as is performed in the Monnet Project[2].

5 Methodological Impact of this Formalism

We have experienced that using a formalism as introduced in Section 2 impacts the role analysts have and the way they work.

Modeling Concerns Separately instead of Consolidating Requirements Classically, analysts spend a lot of time consolidating all the, possibly conflicting, requirements of the different parties involved in a business process of which they are convinced that it meets all those requirements. This takes too much time and introduces problems in the area of traceability: The analyst may be convinced that his process will violate none of the requirements, but this remains implicit in the model. The fact that it meets all requirements follows from the process of modeling, not from the model.

The formalism we introduced allows for a different approach. It is based on the fact that consolidation is left to a computer, and analysts focus on modeling business aspects, in the form of "local" model fragments that reflect a business process from an organizational unit for instance.

This also allows for separation of concerns. More applicative requirements, on how an organization for instance deals with time limits that are about to expire, can be separated from the business requirements on which time limits are to be met within the business process.

Modeling Concerns Separately facilitates Business Ownership The individual activities, and their pre and post conditions, are modeled in relatively modular specifications. Typically, such a local model of requirements maps well to the problem as it is perceived by its owner. For instance, an assessment department might not know or care when an intake form is filled in, but it has no problems expressing the requirement that one is available before assessment can take place. The local model on assessments will now reflect that scope and will express just the requirement.

Focus on Definitions instead of Behavior Classically, business modeling has had an emphasis on the flow across activities, more than on the precise definition of individual activities. The behavior is made explicit, and as a consequence, the definitions often remain implicit. This approach reverses that completely. We now focus on precise and complete definitions of activities, when they may be performed and the consequences of performing them. As a result, definitions are made explicit in the model and the behavior is left implicit. That can however be inferred from the definitions.

Apart from the benefits of flexibility and explainability, this has proven to be useful for our clients in other areas. Agreeing on terminology has helped in

[2] `http://www.monnet-project.eu/`

networked environment to really agree on the processes that are shared and communicate them to all people involved.

Refining a Single Specification instead of Writing Discipline Oriented Deliverables Many methods used in information analysis and design have relied on separating the analysis work into phases and writing separate documents in each phase. An initial requirement analysis might be produced, followed by a functional design. This is a basis for a technical specifications, which in turn is input for realization. Finally, test specifications are written to validate that the system built meets the requirements that were input to the initial requirement analysis. This typically requires hand over between disciplines and translations into discipline related vocabularies.

The approach we have presented allows for a method that is based more on detailing, than on more detailed deliverables replacing the more coarse grained ones. This is important, as change has large impact on the document chains described. A change impacting the requirement document might alter all documents involved, having large impact and introducing traceability challenges. Our formalism allows for expressing more coarse grained choices to be expressed in the same model as the more detailed ones. As a consequence, changes of different impact level can be dealt with in the same models, eliminating the ripple effect. At the same time, the models can be visualized into design views of different abstraction that are appropriate for different phases and responsibilities.

Consistency Checking of Constraints instead of Runtime Processes The fact that our processes are described declaratively impacts the ways the consistency of the processes is assessed. At modeling time, the feasibility of a process is no longer guaranteed by the predictability of an exhaustive definition of flow. Instead, the effects on the processes that may be inferred at runtime when introducing additional constraints may not always be clear up front. As the process is inferred from large numbers of pre and post conditions, (automated) consistency checking is important to detect conflicts. For instance, in cases when cycles of pre conditions occur, no feasible process can be inferred. Also, when an activity's pre conditions, or their recursive pre conditions, are mutually exclusive, an activity can never occur. As the underlying representation is based on a graph, these types of conflicts can be detected by graph traversals at modeling time.

6 Conclusions and Future Work

In this paper we have shown that alternatives to conventional process languages are available today. They address current requirements of enterprises and governments to deal with growing complexity in their product portfolios and a growing demand for agility. The formalism we presented is based on pre and post conditions of activities. The processes inferred from models using this formalism are flexible: We have shown how they allow expert knowledge workers to influence the way they perform their own work, how they can deal well with interventions

and repairs and how they can switch transparently between human and straight through processing.

The formalism supports agile modeling processes with high business user involvement. We have demonstrated the verbalization into natural language, which we have experienced to be a great enabler for business user involvement in enterprise modeling. Additionally, we have presented our experiences on how this type of rule oriented process formalism impacts the methodological approach behind enterprise modeling.

As more of Be Informed's customers adopt this formalism, we would like to do additional research into quantitative aspects of this approach, such as how much it reduces specification size and complexity and how it improves modelers' productivity.

References

1. Business Process Modeling Notation 2.0, `http://www.omg.org/spec/BPMN/2.0/` (Online)
2. Business Process Execution Language 2.0 `http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html` (Online)
3. Le Clair, C., Moore, C., Dynamic Case Management - An Old Catches New Fire, Forrester Research, 2009.
4. Swenson, K.D., Mastering the Unpredictable: How Adaptive Case Management Will Revolutionize the Way That Knowledge Workers Get Things Done, Meghan-Kiffer Press, 2010
5. S. Goedertier and J. Vanthienen: Declarative process modeling with business vocabulary and business rules. In R. Meersman, Z. Tari, and P. Herrero, editors, *On the Move to Meaningful Internet Systems 2007: OTM 2007 Workshops*, LNCS volume 4805, pages 603–612. Springer, 2007.
6. M. Pesic and W. van der Aalst. A declarative approach for flexible business processes management. In J. Eder and S. Dustdar, editors, *Business Process Management Workshops*, LNCS volume 4103, pages 169–180, Springer, 2006.
7. H. Schonenberg, R. Mans, N. Russell, N. Mulyar, and W. Aalst. Process flexibility: A survey of contemporary approaches. In W. Aalst, J. Mylopoulos, M. Rosemann, M. J. Shaw, C. Szyperski, J. L. G. Dietz, A. Albani, and J. Barjis, editors, *Advances in Enterprise Engineering I*, LNCS volume 10, pages 16–30. Springer, 2008.
8. B. Andersson, M. Bergholtz, A. Edirisuriya, T. Ilayperuma, and P. Johannesson. A declarative foundation of process models. In O. Pastor and J. Falcão e Cunha, editors, *Advanced Information Systems Engineering*, LNCS volume 3520, pages 339–377. Springer, 2005.
9. Grondelle, J.C. van, Heller, R., Haandel, E. van, Verburg, T.: Involving Business Users in Formal Modeling using Natural Language Pattern Sentences. In proceedings of EKAW 2010, Lisbon. LNCS volume 6317, Springer, 2010.
10. Kaljurand, K., Fuchs, N.E.: Verbalizing OWL in Attempto Controlled English. In Proceedings of Third International Workshop on OWL: Experiences and Directions, Innsbruck, Austria, 2007.
11. Funk, A., Tablan, V., Bontcheva, K., Cunningham, H., Davis, B., Handschuh, S.: CLOnE: Controlled Language for Ontology Editing. In proceedings of ISWC 2007, Busan, Korea, LNCS volume 4825, pp. 142-155, 2007.