Semantic Business Process Management: Using Semantic Web Services for Business Process Management

Martin Hepp[1, 2], Frank Leymann[3], Chris Bussler[4], John Domingue[5], Alexander Wahler[1], and Dieter Fensel[1, 4]

[1]*DERI Innsbruck, Austria*
[2]*Florida Gulf Coast University, Ft. Myers, FL, USA*
[3]*DERI Galway, Ireland*
[4]*University of Stuttgart, Germany*
[5]*The Open University, Milton Keynes, UK*
martin.hepp@deri.org

Abstract

Business Process Management (BPM) is the approach to manage the execution of IT-supported business operations from a business expert's process view rather than from a technical perspective. The motivation for BPM is that organizations are trying hard to continuously align their actual business processes, as executed by the multiplicity of systems, with the should-be processes as derived from business needs. BPM has gained significant attention by both research and industry, and many BPM tools are already available and in use. However, the degree of mechanization in BPM is still very limited, creating inertia in the necessary evolution and dynamics of business processes, and BPM does not provide a truly unified view on the process space of an organization.

We trace back the problem of mechanization of BPM to an ontological one, i.e. the lack of machine-accessible semantics, and show that the modeling constructs of Semantic Web Services frameworks, especially WSMO [25, 26], are a natural fit to creating such a representation. As a consequence, we propose to combine SWS and BPM and yield one consolidated technology, which we call Semantic Business Process Management (SBPM).

1. Introduction

Business Process Management (BPM) is the approach to manage the execution of IT-supported business operations *from a business expert's process view* rather than from a technical perspective [2, 3]. However, the degree of mechanization in BPM is still very limited, creating inertia in the necessary evolution and dynamics of business processes, and BPM does not provide a uniform representation of an organization's process space on a semantic level, which would be accessible to intelligent queries. In other words, businesses have very *incomplete knowledge of* and very *incomplete and delayed control over* their process spaces.

In this paper, we show that (1) businesses have a need for a unified view on business processes in a machine-readable form that allows querying their process spaces by logical expressions, (2) businesses lack such a machine-readable representation of their process space as a whole on a semantic level, (3) the lack of such a representation is a major obstacle towards mechanization of BPM, and that (4) Semantic Web and Semantic Web services (SWS) technology provide suitable large-scale, standardized knowledge representation techniques. As a consequence, we (5) propose to combine SWS and BPM and yield one consolidated technology, which we call Semantic Business Process Management (SBPM), (6) describe the required components and architecture for SBPM, and (7) outline how this architecture will allow mechanized mediation of the IT / business divide and will thus support both agile process implementation and querying the business process space by logical expressions, e.g. in order to identify activities relevant for compliance with financial or environmental regulations or in emergencies.

1.1. Motivation

The initial motivation for the use of IT in businesses has been the automation of operations, i.e. to mechanize the execution of repetitive transactions. It has for long been common sense to first determine business requirements and then to derive IT implementations – in short, to develop software according to ideal processes as determined by managerial goals. In the early 1990s, Hammer and Champy created the term "Business Process Reengineering" [cf. 8], which brought business

processes to the center of interest and lifted the subject of design from the supporting IT systems to business processes, i.e. to the perspective of business experts. This initiated a wealth of contributions from academia and practitioners, which also stimulated the development of Business Information Systems research as a scientific discipline. However, the popularity of Business Process Reengineering did not change the underlying sequential paradigm of (1) analyzing the current state extensively, (2) yielding the description of an improved state, and (3) modifying existing systems in an engineering-fashion to implement the necessary changes.

This strict sequential model of IT design in enterprises, however, has led to enormous problems, because organizations as living systems are in continuous change, which means that every requirements analysis can become partly outdated while we are working on the implementation in the next stage of the systems engineering process, and the longer the cycles take, the more a problem this becomes. This was a lesser issue when (1) the use of IT in organizations was limited (there were little "legacy" systems), when (2) market structures were more stable, and when (3) the level of integration with suppliers and customers was low. Nowadays, however, organizations are trying hard to continuously align their actual business processes, as executed by the multiplicity of systems, with the should-be processes as derived from managerial needs. It is a common observation that the launch of a new product or the implementation of a new revenue scheme for the employees will be determined by the ability to set up the required processes within the existing IT landscape.

If companies are to survive in a dynamic environment, they are subject to competition in at least three dimensions (see Figure 1): Cost per process execution (y-axis), cost per process setup (z-axis), and delay of process setup (x-axis in the figure). Thus, they have to aim at minimizing theses three dimensions. In other words, companies must be *efficient* (low cost per transaction in the operational stage), *agile* (low lag in setting up new or modified processes), and *able to evolve their process space in small iterations* based on low lead costs for setting up new or modified processes. Such enterprises that meet these requirements operate well and reside in the dashed cube in the figure. If, on the contrary, the cost per transaction is too high, there is a lack of efficiency (space above the cube); if it takes too long to set up a new process, there is a lack of agility (space right to the cube), and if the lead cost for setting up a new process is too high, the organization is unable to set up processes for small business opportunities or minor improvements (space behind the cube).

Now, although a significant part of the process execution is already stored in computer systems (e.g. in the form of code fragments, data structures, data, system links, etc.), both querying and manipulating the process space regularly requires human labor. Obviously, there is a functional bottleneck between (1) the *business perspective* on operations and (2) the *actual execution* of operations on IT systems [3]. Figure 2 illustrates this IT/business divide:

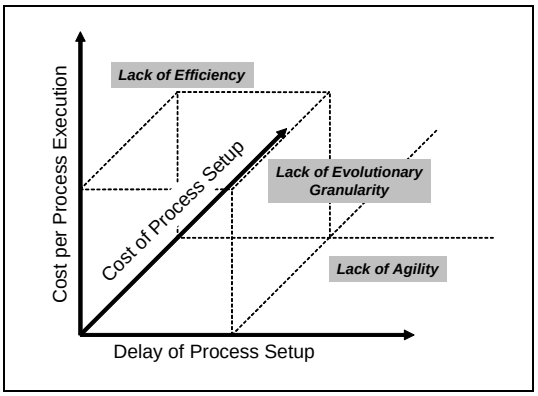

Figure 1. The three dimensions of enterprise performance from a process space view

The upper triangle depicts the perspective of business experts; the lower triangle represents the actual implementation, which includes all computer systems and man-machine teams. The transition between those two spheres is very narrow, as there is no automated mediation between them. In other words, *the fundamental problem is that traversing from one sphere to the other requires manual labor in any of the two directions, i.e. both for querying and manipulating the process space:* If a manager needs to know all billing processes, systems analysts have to try to create an inventory of any such processes; and if a manager needs a new billing process for a new product or service, software engineers have to transform the management requirements into an IT implementation. This leads to the situation that business-process-related activities are, amidst a wealth of IT, surprisingly centric to human labor, and thus slow, costly, and imperfect.

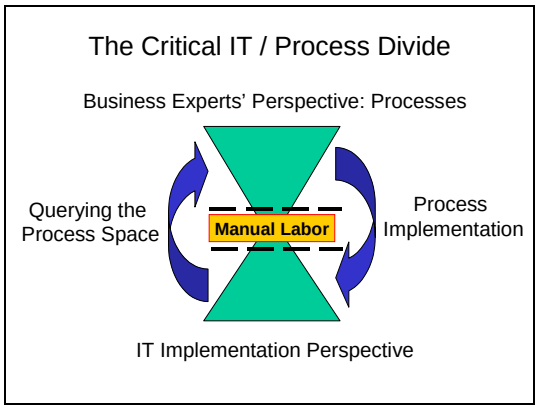

Figure 2. The bi-directional IT / process divide (derived and extended from http://www.bpmi.org)

This gap has been targeted by the emerging field of Business Process Management (BPM) [2, 3]. BPM aims at providing tools and techniques that support the modeling, management, and monitoring of operations on a business process level, while automatically mapping this high-level perspective to the actual implementation being executed on the multiplicity of systems. BPM tools usually put a strong emphasis on the graphical representation of processes, augmented with workflow and Enterprise Application Integration (EAI) functionality. In brief, BPM is a promising new area that provides a high-level perspective on business processes inside an organization. However, its current implementation does not overcome the underlying limitation that *the business process space inside the organization as a whole is not accessible on a semantic level*, especially because business process modeling languages like BPEL4WS [1] are an insufficient means of capturing and representing such a domain of discourse.

In our opinion, Business Process Management will only fulfill its promise if it provides full mechanization support for traversing the IT/business divide in both directions, e.g. answering queries like "Can we set up a billing process that completes in less than 0.3 seconds and costs less than $0.10 per transaction?" or enacting new process instances according to a machine-readable representation of a *goal*, and not only according to representations of a *process orchestration* as in BPEL4WS. The major obstacle is that both the business experts' perspectives on business processes and the IT implementation sphere are *widely not accessible at a semantic level and thus to machine reasoning*. Only this will help organizations achieve the desired effectiveness, agility, and ability to exploit small opportunities – in other words, to be located inside the target cube of Figure 1.

Semantic Web Services (SWS) aim at mechanizing the discovery and composition of Web services and provide means for the representation of executable artifacts that are accessible to intelligent queries and machine reasoning. In the following, we will show that BPM is a natural application domain for SWS, and that the modeling elements of the WSMO framework cover the representational needs of SBPM much better than existing process modeling languages like BPEL4WS.

1.2. Approach

Semantic Web technology, namely ontology languages, repositories, reasoners, and query languages, provides scalable methods and tools for the machine-accessible representation and manipulation of knowledge. Semantic Web Services (SWS) make use of Semantic Web technology to support the automated discovery, substitution, composition, and execution of software components, namely Web Services. Our idea is to combine SWS and BPM, and to develop one consolidated technology, which we call Semantic Business Process Management (SBPM). SBPM takes the following approach: We

(1) represent and semantically describe each existing atomic and composite process inside an organization as a SWS in a process repository;
(2) capture the complete IT landscape (e.g. hardware, operating systems, manufacturing equipment) in the form of an ontology;
(3) gather domain knowledge (e.g. technical constraints, business rules) and store it in the form of axioms in a rule language (which can be part of the ontology language);
(4) map transactional data from the various systems (e.g. ERP) to an instance store;
(5) express queries in an ontology query language;
(6) model business experts' needs as WSMO goals, and
(7) use a SWS execution environment for the mediation between business goals and queries, and the actual process space.

The structure of this paper is as follows: In section 2, we identify the two ways of accessing the process space of an organization as either querying or manipulating this space, and present brief use cases for those two types of actions. In section 3, we derive from these use cases the required functionality for SBPM and partition it into five fundamental sub-problems. Additionally, we transform the requirements from the use cases into requirements for

those sub-problems. In section 4, we analyze the contribution of Semantic Web Services frameworks, namely WSMO [25] to a solution of these sub-problems. In section 5, we discuss our findings, and assess the advantages of SWS frameworks to BPM as compared to existing process representation languages. Section 6 summarizes our findings.

1.3. Related Work

Our work is related to the following research directions:

Workflow management: For an overview of production workflow management including the role of business processes and their modeling, see for example [15]. [17] discusses the impact of using workflow technology on the creation of applications. [18] describes an overall environment for modeling, testing, deployment, running, analyzing applications based on business processes (i.e. the lifecycle).

Business Process Management: The vision of BPM is outlined in [2, 3]. [9] sketches the role of business processes as an artifact in software engineering. [19] discusses the use of business processes in cross-enterprise interactions. [20] positions Web services and business processes as the basis for future application structures.

Semantic Web Services: OWL-S [13] is a comparatively narrow framework and ontology for adding semantics to Web service descriptions. WSMF is a more comprehensive framework [23], which has been further developed to the Web Service Modeling Ontology (WSMO). The core specification of WSMO can be found in [25], a brief introduction is given in [26]. WSML [24] is a family of fully-fledged ontology representation languages that supports WSMO. IRS-III [10] and the Web Service Execution Framework WSMX [27] are two reference implementations of WSMO.

SWS are currently subject to intense research, especially in the DIP project[1], and it is thus outside the scope of this paper to summarize all related work.

Business Process Modeling Languages and Standardization Initiatives: The BPM and Web Services communities have yielded a wealth of languages and standardization approaches that aim at describing business processes, especially from the perspective of Web Services orchestration. The most prominent examples are BPEL4WS [1], BPML [4], BPMN [6, 7], XLANG [30], WSCI [22], and WS-CAF [21]. In short, all those focus only on the representation of a narrow part of the process space, namely the patterns of message exchange (choreography) and the control flow in the combination of multiple Web services (orchestration).

Mining the process space: One major challenge towards the vision of SBPM is automatically capturing the process space. There are at least two earlier works that can be build on. Reverse Business Engineering [14] is a methodology and family of toolsets that read out transactional data and program module usage in ERP systems, namely SAP R/3 and mySAP, in order to analyze the process space of an organization. Additionally, [11] describes the usage of data mining technology for deriving process models form historical information, i.e. a new kind of analysis technique in the business process lifecycle.

2. Usage Scenarios

In this section, we discuss accessing the process space of an organization, and present brief use case scenarios for such types of actions. We argue that all forms of management tasks related to the process space of an organization can be traced back two just two fundamental types of usage, i.e. either querying or manipulating this space.

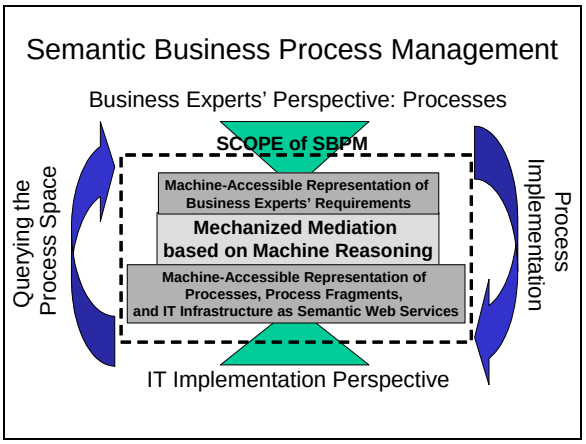

Figure 3. Semantic Business Process Management

2.1. Querying the Process Space

In management science, decision making is a core discipline, and the main challenge for good decision making is having access to all required information. This might sound like a triviality, but in fact reveals that querying the process space is a very important task. We understand a query as (1) a machine readable

[1] http://dip.semanticweb.org

representation of (2) a logical expression that (3) defines a subset of all facts ("knowable" things might be more appropriate) in the universe of discourse (i.e. the process space) and (4) is used as request for returning all known facts that match this logical expression.

We envision the following examples of queries as reflecting very typical managerial information needs:

- *"List all business processes that depend on system x."*
- *"Do we have a cost approval process for items below $ 200?"*
- *"Do we have inter-organizational processes that involve company y?"*
- *"How many business transactions do we carry out with partner z on an average day?"*
- *"How many inventory management methods are currently in use?"*
- *"Can we compose a billing process model that complies with the attached specification of elements and control flow and costs less than $ 0.1 per transaction?"*
- *"In which of our food manufacturing machines are we processing meat or raw eggs?"*

Such queries can be time-critical in order to identify activities relevant for compliance with financial or environmental regulations or in emergencies. One can easily see that such types of queries cannot be supported by syntactical process standards or simple databases, but that fully-fledged knowledge representation techniques are needed. An obvious reason for this need is that a huge part of the facts needed to answer such queries will be *implicit* information. For example, we might have a database of all food processing machinery and this might even contain the type of food processed, but this does not allow searching for generalizations (e.g. "Microsoft OS" as the super-category of various versions of MS Windows) or symmetrical relationships (if we know that system 1 is connected with system 2, then we implicitly know that system 2 is also connected with system 1).

The ability to answer such queries spanning the whole process space requires

(1) a machine-accessible representation of all relevant facts (concepts, instances, and axioms) on the implementation and execution level and

(2) a machine-accessible representation of the queries.

On first view, pure Semantic Web techniques, namely ontologies, repositories, and reasoners, are sufficient. In other words, it would be sufficient to "ontologize" the process space. However, a very important type of queries are in the form "can we enact / compose a process that does xyz?". This is a typical SWS discovery and composition scenario.

2.2. Manipulating the Process Space

The second form of accessing the process space is manipulating it. Examples are to create a new business process, modify an existing process, or shut down an outdated process

We envision the following examples of requests as reflecting very typical managerial process space manipulation needs:

- *"Compose a process model that complies with the attached specification of elements and control flow if possible; if not, return the gaps."*
- *"Compose a process model that achieves the attached goal."*
- *"Set up a billing process model."*
- *"Create a billing process instance."*
- *"Replace process fragment A in all processes by process fragment B, if doable."*
- *"Execute process A every time process B is executed (completed, terminated)."*

Such functionality requires the same representations as described in section 2.1 plus at least

- the ability to actually invoke the represented functionality, e.g. via Web service calls,
- a component that can resolve any given request into an orchestration, and
- a workflow engine than can execute the resulting orchestration.

Basically, current BPM techniques and business process languages can cover part of these requirements. For example, one can define the orchestration of a business process in BPEL4WS and pass this to an execution environment which will actually enact and execute this process. However, we do not only want to enact processes for which we already have the description of the orchestration and for which we know the components (and know that they are available), but we also want to be able to describe goals and leave it to the BPM environment to figure out whether and how this can be implemented.

3. Requirements and Sub-Problems

Achieving the described level of business process management automation can be broken down to five sub-problems, i.e.

- discovery of facts,
- representation,
- query and retrieval,
- enactment and execution, and

- mediation, i.e. the mechanized resolution of interoperability problems caused by heterogeneities.

Figure 4 shows the architecture of an SBPM environment.

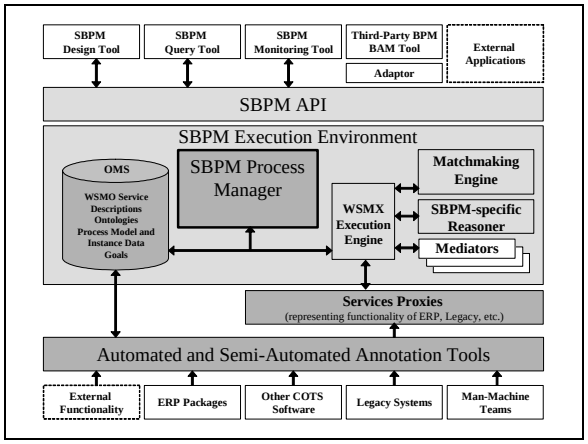

Figure 4. Simplified Architecture of an SBPM Environment

3.1. Discovery of Facts

The very first step and probably the most difficult one is discovering and annotating all facts in the universe of discourse. We need to analyze all systems, software, and data in order to collect the following facts:

(1) The complete IT landscape (e.g. hardware, operating systems, manufacturing equipment, technical constraints,…);
(2) the business logic, e.g. existing atomic and composite process inside the organization and business rules;
(3) man-machine teams and human labor;
(4) transactional data from the various systems (e.g. ERP), and
(5) environmental parameters.

The basic challenge in making this a reality is the development of powerful crawlers and annotation tools, which can build upon work done in the field of mining process models form historical information [11] or the Reverse Business Engineering approach as applied by the RBE family of tools [14].

3.1.1. IT Implementation Level

We need a comprehensive representation of the IT landscape, including

- all infrastructure components, like systems, components, capabilities;
- all application packages, e.g. “SAP myERP”, “Oracle Financials”;
- proprietary software solutions and legacy systems;
- hard- and software requirements of application packages;
- requirements of process models inside application packages; and
- low-level services.

3.1.2. Business Logic Level

At the next level, we need to capture all process models and process model fragments, which are mainly hidden inside COTS application packages or the control flow in proprietary software. Additionally, we need to capture business rules that reflect the constraints and dependencies between process models. Three core examples are

- *supported* process models both inside COTS application packages and legacy systems, e.g. “Kanban-based reorder”, “Inventory management based on forecast consumption”, “Min-Max reordering”, “Vendor-managed inventory”;
- *enacted* process models inside COTS application packages and legacy systems; and
- management science domain knowledge.

Huge ERP systems are nowadays libraries of best-practice processes in most areas of business operations and management, and usually come with an exhaustive documentation. In this case, deriving the process logic and facts will be a lot easier than with proprietary software, and also economically a lot more reasonable, since the results will be useful for a large group of users.

3.1.3. Man-Machine Teams and Human Labor

Only part of the process models and process model fragments are hard-coded in computer software. The other part is implemented by human workforce. Such processes and process fragments need also be captured. A starting point for the discovery of such facts is a skills database, mining interaction patterns (e.g. e-mail headlines), tool-support for self-description, and logging data of program usage (e.g. the one who uses the general ledger application twice a week can be assumed to be a bookkeeper). Also, the constraints of human labor (physical, legal, social,…) need to be collected.

3.1.4. Execution Level

Most data on the execution level resides in the transactional databases of the organization. Transactions stored in such databases reflect process

instances of IT-supported process models. Due to the huge amount and volatile nature of the data, a bulk important into an ontology repository seems to be unfeasible. On the other hand the data is easily accessible. The challenge will be to persistently store data from the execution level in a data warehouse and make this data warehouse accessible as an ontology instance store by transforming the warehouse metadata into fully fledged semantic annotations.

3.1.5. Environment

Last, also the environment of the organization needs to be analyzed. Available machinery need to be inventoried and parameters and constraints of the environment need to be captured (e.g. capacity of utility supplies, legal constraints and regulations,…).

3.2. Representation

The previous section has outlined the elements of the BPM universe of discourse. In this section, we discuss the representation for the discovered facts. Current business process modeling approaches like BPEL4WS [1] or BPMN 1.0 [6, 7] focus on just a few aspects of business processes. BPEL4WS, for example, is mainly a standard syntax for describing the orchestration of a business process, whereas BPMN is a graphical notation for describing the control flow of a business process in a form suitable for business experts; it also provides a mapping of such diagrams to BPEL4WS. However, none of the existing languages provides the expressiveness and degree of formal semantics necessary for the representation of the facts discovered in section 3.1.

3.2.1. Required Modeling Primitives

If we want to represent the facts from section 3.1, we need at least the following modeling primitives:

(1) Named classes (concepts), e.g. “Kanban-based inventory management”, “billing”, “invoice”, etc.
(2) Instances, e.g. one specific invoice.
(3) Instance-class relationships, so that we can represent that a specific invoice is an instance of an invoice.
(4) Class-subclass relationships, so that we can represent that every instance of “Windows XP” is a also an instance of “Microsoft Operating System”.
(5) Relations that can be used to express relationships between classes or instances and the possibility to express properties of these relations like transitivity and symmetry (i.e. to define axioms in a rule language).
(6) Axioms, i.e. logic expressions that reflect domain knowledge.
(7) Data type definitions as a prerequisite for operations on literal values, especially conversion, comparison, and arithmetic operations.

One can already see that fully-fledged ontology languages like WSML have a very good fit to those needs, and that XML syntax standards have not.

3.2.2. Required Vocabulary

Provided that suitable modeling primitives are available, the following vocabulary has to be developed and agreed upon in various communities. It is important to note that there will be a multiplicity of such vocabularies which will all have only partly overlapping communities of support. From a knowledge engineering perspective, all those are ontologies or parts of ontologies.

(1) Industry-specific set of consensual concepts, e.g. all the concepts used in the telecommunications business domain (“router”, “leased line”, “modem”), sufficiently augmented by axioms.
(2) Generic set of consensual classes, e.g. all the concepts of business (“invoice”, “customer”, “key customer”, “credit card”, also augmented by axioms.
(3) Industry-specific consensual set of relations, e.g. identifiers and definitions for the possible relations between concepts, between concepts and instances, and between instances, e.g. in the rental business “is leased to”, “is maintained by”. The relations are also to be augment by axioms.
(4) Generic set of consensual relations; as above, but industry-neutral like “paid by”.

3.2.3. Required Axioms

On all levels of discovery we also have to capture the rules of the domain in the form of axioms. As this is a very wide area, we need a fully-fledged rule language for this purpose.

3.2.4. Repository

There must be a repository for the persistent storage of all components of the representation. For the storage of instance data, a virtual instance store in the form of a mapping to a data warehouse, to be augmented with a semantic annotation of the metadata, may be a promising approach.

3.2.5. Process Language

We need a language that can represent all aspects of a process model and process instances, i.e.

- the choreography of a process (i.e. its pattern of interaction with the outside world),
- the orchestration of a process (i.e. how it invokes other services in order to achieve its functionality; this is needed in order to capture the dependencies, e.g. for checking whether input services are available),
- the preconditions of a process (i.e. its information space prior to its execution),
- the assumptions of a process (i.e. the necessary state of the world prior to the execution of the process),
- the postconditions of a process (i.e. the information space after its execution),
- the effects of a process (i.e. the state of the world after to the execution of the process) [cf. 25, 26].

3.2.6. Query Language

In order to express queries to the process space, we need a query language that supports all the modeling primitives of the chosen representation. Also, convenient features known from database query languages like SQL (e.g. nesting) are beneficial.

3.3. Query and Retrieval

With regard to querying process spaces, Klein and Bernstein [15] have proposed a Process Query Language (PQL) and were able to show that process-based representation and queries of services result in a much higher precision and recall than keywords- and table-based approach. They also claimed that concepts-based retrieval (ontology-based) and deductive (expressing service semantics formally using logic) are too complex or prohibitively difficult, which we do not take for granted. Instead, we assume that "ontologizing" the process space of an organization by using automated tools is possible and, quite the opposite, in the long run the only feasible way of managing the process space of a business. Under this assumption, processing queries on the process space is no different from querying an ontology repository, and we expect that available reasoners can be employed for this purpose.

3.4. Mediation

The process space of an organization will undergo changes very different paths of evolution, varying by department, type of activity, type of systems involved, etc. This will quite naturally result in interoperability problems caused by heterogeneities, e.g. on the data or process level. An example for such heterogeneities on the data level are the usage of different identifiers for the same concept, and one of the process level is if one process is used to send a user name and a password in one turn and the other process expects them one by one with intermediate acknowledgements.

If we want to increase the degree of automation in general, it seems important to provide software components that can help overcome occurring interoperability conflicts and this in an automated fashion. This functionality is known as mediation and the respective components are called mediators. Mediation can take place on a multiplicity of levels, e.g. on the level of data, ontologies, processes, protocols, or goals. To a great extend, it will depend on the availability of sophisticated and industry-strength mediation support whether the promise of a Semantic Web services can become a reality, which is also reflected in the fact the mediators are a core component of the Web Service Modeling Ontology (WSMO, [25, 26]).

3.5. Enactment and Execution

Solving all previous sub-problems would make the process space accessible to intelligent queries, but would not be sufficient for actually manipulating the process space from an business experts' perspective. The later requires additionally

- a component that can resolve any given request into an orchestration,
- the ability to modify or delete existing process models (mainly orchestrations),
- a workflow engine than can execute the resulting orchestrations and can actually invoke the represented functionality, e.g. via Web service calls, and
- the ability to enact (instantiate) and execute a process model.

Please note that even semi-automated solutions will be very beneficial, though full mechanization is the long-term goal. For example, man-machine teams can be represented using dummy Web services that just inform the relevant individuals of the task. So it is even possible to include manual tasks in mechanized execution of an orchestration.

4. Suitability of Semantic Web Services Frameworks for BPM

In this section, we analyze the contribution of Semantic Web Services frameworks, namely WSMO [25] to a solution of the sub-problems identified in the previous section.

4.1. Discovery of Facts

WSMO is a conceptual framework for Semantic Web services. As such, it provides the required modeling constructs, but does not address implementation details. Thus, the sub-problem of creating automated and semi-automated annotation tools for the discovery of facts, as outlined in section 3.1, is an open research question. However, previous work from the Semantic Web research community in the field of annotation will very likely contribute to a solution.

4.2. Representation

As shown in Table 1, WSMO v1.2 provides all modeling primitives from the requirements analysis, whereas BPEL4WS lacks most of them. One has to admit that BPEL4WS was also not designed for this purpose, but since it is frequently cited as a representational means for processes, it seems important to highlight its limitations.

Table 1. Availability in required modeling primitives in BPEL4WS and WSMO

Modeling Primitive	BPEL4WS v1.1	WSMO v1.2
Named classes (concepts)	Yes	Yes
Instances	No	Yes
Instance-class relationships	No	Yes
Class-subclass relationships	No	Yes
Relations	No	Yes
Axioms	No	Yes
Data type definitions	Yes (XSD)	Yes (in WSML)

With regard to yielding consensual vocabularies and capturing axioms of the domain of discourse, WSMO as a conceptual framework does not help, since creating ontologies by engineering means or semi-automatically and discovering axioms is outside of the scope of WSMO.

The reference implementations of WSMO, WSMX and IRS-III, include own or reuse existing ontology repositories and thus cover the need for a repository. It remains an open question whether those prototype do already scale sufficiently.

The requirements on a process language as imposed in section 3.2.5, i.e. orchestration, choreography, preconditions, assumptions, postconditions, and effects are met by WSMO v1.2, whereas only orchestration and choreography issues are covered by BPEL4WS v1.1.

4.3. Retrieval

WSMO provides a rich conceptual model for Web service discovery based on the separation of goals (functionality for which fulfillment is sought) and Web services and their capabilities. For more details, see http://www.wsmo.org. In a nutshell, both the query for facts including reasoning support and the comprehensive discovery of processes is covered by WSMO v1.2.

Currently lacking is dedicated support for gap analysis, e.g. retrieving the pieces of functionality missing for achieving a specific goal.

4.4. Mediation

Yet the Web Service Modeling Framework (WSMF, [23]) included mediators as a core components. WSMO v1.2 explicitly defines four types of mediators, i.e. ooMediators (ontology-ontology), ggMediators (goal-goal), wgMediators (Web service-goal), and wwMediators (Web Service-Web Service) [25]. From a conceptual standpoint, this seems to cover the core needs for mediation. However, the implementation of comprehensive, efficient, and scalable mediators of industry-strength is still in its infancy. One should admit, though, that other frameworks (e.g. OWL-S, [13]) do not even cover this need sufficiently on a conceptual basis.

4.5. Execution

The two explicit needs of an SBPM execution engine are not covered by WSMO, but the two reference implementations WSMX [27] and IRS-III [10] can be expected to serve as an important core of the functionality.

5. Discussion

Business Process Management has already yielded several sophisticated tools (e.g. Ultimus BPM Suite or Savvion BusinessManager) that exceed simple graphical workflow design approaches at the level of Microsoft Visio. Those tools usually contain a design environment, in which business experts can create or modify business processes and specify business rules. This is usually accompanied by a BPM repository containing technical details of the existing IT components and processes, and a BPM engine that coordinates the execution of the individual process steps by triggering the involved applications. Some packages additionally include Business Activity Monitoring (BAM) functionality that helps business experts monitor the execution performance of processes. The working group BPMI.org has started the development of a BPM standards stack, with the Business Process Modeling Notation (BPMN, [6, 7]) as the most progressed component. BPMN is based on the OMG's UML 2.0 Diagram Interchange Specification and provides a means of expressing business processes for business experts using a graphical notation. However, the BPMI standards stack is based on traditional Web Services standards, namely WSDL and UDDI as representation means for the implementation level (i.e. the functionalities that are available for the actual execution). Thus, BPM in its current stage inherits all the limitations of the traditional Web Services stack, namely UDDI as an insufficient approach to discover Web Services automatically. Thus, the brittleness of current Web Service technology, caused to a large extent by the lack of automated mediation, remains. In addition to that, current BPM is not founded on expressive, logic-based representation techniques, and thus fails at making the whole business process space accessible to intelligent queries and machine reasoning.

In short, the insufficient degree *of machine-accessible representations of the processes and data about processes* inside organizations creates the following problems in current Business Process Management:

(1) **Low degree of automation in the implementation stage:** The actual setup of business processes according to managerial needs is mainly done manually, often involving numerous consultants.

(2) **Implementation delay:** The dynamic composition of business processes is mostly impossible, increasing the time to market and reducing an organization's agility.

(3) **Cognitively inadequate complexity:** The lack of a clear separation between business goals and implementation details makes the management of business processes overly complex.

(4) **Process blindness:** Managers and other business experts cannot quickly determine whether a specific process can be composed out of existing atomic processes, nor can those stakeholders query the process space within their organization by logical expressions. Thus, checks for process feasibility (e.g. prior to the launch of new products or services) or compliance (e.g. Sarbanes-Oxley, ISO, etc.) are still to be done manually by business analysts.

(5) **Intransparent concurrency conflicts and interdependencies:** It is impossible to use machine reasoning in order to identify potential side-effects of modifications. Also, process improvement should strive for a global process optimum, not local process optima; however, this cannot be achieved without a proper representation of interdependencies.

We have yielded the representational and functional needs for true mechanization of business process management and mapped those requirements to both BPEL4WS v1.1 and WSMO v1.2. This clearly shows that BPEL4WS covers only a very small part of the requirements for a comprehensive representation of the process spaces of organizations. In contrast, WSMO shows a natural fit to the requirements.

As WSMO is a conceptual model, it remains outside the scope of WSMO to address several implementation issues, especially with regard to automatically capturing the process space in all of its details. However, one can say that if the facts can be yielded, then they can be represented using WSMO and its representation language WSML. Also, as WSMO builds on top of the research carried out in the Semantic Web community, we can expect to successfully reuse available implementation results.

6. Conclusion

We have shown that businesses lack a machine-readable representation of their process space as a whole on a semantic level, while they would benefit from such a unified view on business processes in a form that allows querying their process spaces by logical expressions. We also substantiated that the lack of such a representation is a major obstacle towards mechanization of BPM, and that Semantic Web and Semantic Web Services (SWS) technology provide suitable large-scale, standardized knowledge

representation techniques. As a second step we proposed to combine SWS and BPM and yield one consolidated technology, which we call Semantic Business Process Management (SBPM), described the required components and architecture for SBPM, and outlined how this architecture will allow mechanized mediation of the IT / business divide and will thus support both agile process implementation and querying the business process space by logical expressions, e.g. in order to identify activities relevant for compliance with financial or environmental regulations or in emergencies.

We are currently in the process of creating a WSMO/WSMX-based use case and proof of concept for the telecommunications industry, and are working on a comprehensive stack of standards for SBPM.

References

[1] T. Andrews et al.: "Business Process Execution Language for Web Services Version 1.1". Available at http://www.siebel.com/bpel.

[2] H. Smith, P. Fingar: "Business Process Management. The Third Wave. Meghan-Kiffer Press, Tampa, FL, USA 2002.

[3] Business Process Management Initiative, http://www.bpmi.org.

[4] A. Arkin: "Business Process Modeling Language", November 13, 2002. Available at http://www.bpmi.org/downloads/BPML1.0.zip.

[5] BPMI.org: "BPML / BPEL4WS. A Convergence Path toward a Standard BPM Stack", Position Paper, August 15, 2002. Available at http://www.bpmi.org/downloads/BPML-BPEL4WS.pdf.

[6] BPMI.org: "Business Process Modeling Notation (BPMN) Version 1.0". Available at http://www.bpmi.org/downloads/BPMN-V1.0.pdf.

[7] S. White: "Introduction to BPMN". Available at http://www.bpmi.org/downloads/Introduction_to_BPMN89.pdf.

[8] M. Hammer, J. Champy: "Reengineering the Corporation", Nicholas Brealey Publishing, 2001.

[9] F. Leymann: "On the interrelationship of workflow technology and other software technologies", in: Proc. 5th Intl. Conf. on the Software Process ICSP5 (ACM SIGSOFT), Chicago, Illinois, USA, June 15-17, 1998.

[10] S. Galizia, J. Domingue: "Towards a Choreography for IRS-III", in: Proceedings of the Workshop on WSMO Implementations (WIW 2004) Frankfurt, Germany, September 29-30, 2004.

[11] R. Agrawal, D. Gunopulos, F. Leymann: "Mining process models from workflow logs", in: Proc. Intl. Conf. on Extending Database Technology EDBT'98, Valencia, Spain, March 3-8, 1998.

[12] W3C: "OWL Web Ontology Language Reference". W3C Recommendation 10 Feb 2004, 12 November 2002. Available at http://www.w3.org/TR/owl-ref/.

[13] OWL-S 1.0 Release, http://www.daml.org/services/owl-s/1.0/.

[14] IBIS Prof. Thome AG: „RBE Plus". Available at http://www.rbe-online.de.

[15] M. Klein, A. Bernstein: "Toward High-Precision Service Retrieval", IEEE Internet Computing Vol. 8, No. 1, 2004, pp. 30-36.

[16] F. Leymann, D. Roller. "Production Workflow - Concepts and Techniques", PTR Prentice Hall, 2000.

[17] F. Leymann: "The workflow-based application paradigm", in: Proc. Workshop on Workflow Managment - State of the Art, Münster, Germany, April 10, 1996.

[18] F. Leymann, D. Roller: "Workflow based applications", IBM Systems Journal 36(1) 1997, pp. 102-123.

[19] F. Leymann, D. Roller, M.-T. Schmidt: "Flows and Web Services: B2B aspects of business process management", IBM Systems Journal 41(2) 2002.

[20] F. Leymann: "The Influence of Web Services on Software: Potentials and Tasks", in: Proc. of the 34th Annual Meeting of the German Computer Society, Ulm, Germany, September 20 – 24, 2004.

[21] Web Services Composite Application Framework (WS-CAF). Available at http://developers.sun.com/techtopics/webservices/wscaf/.

[22] W3C: "Web Service Choreography Interface (WSCI) 1.0", W3C Note 8 August 2002. Available at http://www.w3.org/TR/wsci/.

[23] D. Fensel, C. Bussler: "The Web Service Modeling Framework WSMF", Electronic Commerce Research and Applications, 1 (2) 2002.

[24] J. de Bruijn et al.: "D16.1v0.2 The Web Service Modeling Language WSML", WSML Final Draft March 20, 2005. Available at http://www.wsmo.org/TR/d16/d16.1/v0.2/20050320/

[25] D. Roman et al.:"D2v1.2 Web Service Modeling Ontology (WSMO)", WSMO Final Draft April 13, 2005. Available at http://www.wsmo.org/TR/d2/v1.2/20050413/.

[26] C. Feier, J. Domingue: "D3.1v0.l2 WSMO Primer", WSMO Working Draft April 1, 2005. Available at http://www.wsmo.org/TR/d3/d3.1/v0.2/20050401/.

[27] Web Service Execution Environment (WSMX). Available at: http://www.wsmx.org/

[28] S. Thatte: "XLANG. Web Services for Business Process Design". Available at http://www.gotdotnet.com/team/xml_wsspecs/xlang-c/default.htm.