

Discovering Patterns for Inter-Organizational Business Collaboration in a Top-Down Way

A. Norta and P. Grefen

Eindhoven University of Technology, Faculty of Technology and Management, Department of Information Systems, P.O. Box 513, NL-5600 MB, Eindhoven, The Netherlands.
a.norta@tm.tue.nl

Abstract. In the area of business-to-business (B2B) collaboration, original equipment manufacturers (OEMs) are confronted with the problem of spending a considerable time and effort on coordinating suppliers across multiple tiers of their supply chain. In tightly integrated supply chains the failure of providing services and goods on time leads to interruptions of the overall production and subsequently results in customer dissatisfaction. This paper proposes the concept of electronic *Sourcing* as a new approach for improving the coordination of service provision across several tiers of a supply chain. Sourcing allows for the harmonization of heterogeneous system environments of collaborating parties without requiring a total disclosure of internal business details to the counterpart. Furthermore, with tool support in Sourcing it is possible to verify the correct termination of processes and the contractual adherence of service provision without imposing fixed standardized routing. As Sourcing is a new proposal for B2B supply-chain collaboration, this paper analyses features of the Sourcing concept in a pattern-based way. However, differently to intra-organizational perspectives such as control-flow, data-flow, or resource, where established in-house workflow systems and web service composition languages (WSCLs) were analyzed for patterns, this paper pursues an analysis of Sourcing patterns in a top-down way. The reason for proceeding top-down is a lack of available systems for B2B collaboration that have gone through a lengthy period of adoption to real world business activities. The discovered and specified Sourcing patterns of this paper are instrumental in the EU-FP6 project CrossWork for the conduction of case studies with industry partners from the automobile industry.

1 Introduction

After exploring and successfully applying workflow concepts for intra-organizational applications [6, 31], enterprises in the B2B domain are faced with the next challenge of achieving increased efficiency and effectiveness in the area of inter-organizational collaboration for commercial exchanges. Traditionally, a business transaction starts with negotiating and defining a contract that contains essential elements like the clear specification of a service that needs to be provided by one contracting party and how much compensation is offered by another contracting party that consumes this service. The parties involved in the contract have their own internal processes that need to be aligned inter-organizationally for the duration of the enactment of a business transaction. However, collaborating companies are confronted with many problems. For example, the

business processes of the respective parties are supported by systems that are based on concepts that differ extensively and therefore inter-organizational collaboration is too cumbersome or not possible at all. One party uses a process-formulation language that contains constructs for which no comparable constructs exists in the domain of the collaborating counterpart. The situation may occur that processes terminate correctly in-house but deadlock when they are connected because of a data-flow or control-flow problem. Furthermore, setting up an inter-organizational business process collaboration is time-consuming and costly. Thus, it is desirable to have web-based middleware and WSCLs available for an automated alignment of inter-organizational processes so that the B2B transactions are harmonized in an effective and efficient way.

This need for middleware and languages that support inter-organizational collaboration is under investigation by different research communities [15, 28, 29, 50]. Summing up these efforts, the research direction is on the one hand focusing on exploring workflow concepts and on the other hand including service-oriented business integration (SOBI). These two aspects are united in the framework of dynamic inter-organizational business process management (DIBPM) [21] where the need of organizations is addressed for dynamically bringing together the processes of a service consumer and provider over web-based infrastructures. In this case *dynamically* means that organizations are found and matched just-in-time in an instance based way. In the domain of SOBI, web service composition languages (WSCL) have emerged for supporting process specifications, e.g., BPEL, BPML, WSCI, WSFL, XLANG, and so on [17–19, 25, 45]. However, while these WSCLs succeed in choreographing the execution of composite services, they also require enhancement and harmonization for meeting the business requirements of collaborating parties.

Since existing applications and languages support DIBPM only partially, the question arises how to explore DIBPM features without being overwhelmed by the complexities resulting from inherent business issues, concepts, and heterogenous technologies. Thus, it is essential to find an approach that results in a separation of concerns so that a structured investigation takes place. Intra-organizational workflow systems have been explored [7] in different perspectives by using a pattern-based analysis approach. Accordingly, this paper also proposes a pattern-based exploration for DIBPM. However, in the latter case the challenge occurs that such an exploration can't be carried out in a bottom-up way by analyzing inter-organizational applications and WSCLs that have proven themselves over many years and been adjusted to real-world problems. For DIBPM such "reality-proven" systems and WSCLs are not yet available. Therefore, this paper pursues a pattern-based exploration of DIBPM in a top-down way, which leads to a catalog of patterns that are for harmonizing the business processes of parties that intend to engage in a business collaboration.

The structure of the paper is as follows. First, Section 2 presents related work and Section 3 briefly introduces the concept of Sourcing. Next, control-flow properties of Sourcing are informally described in Section 4 followed by a presentation of Sourcing dimensions in Section 5 that are relevant for a top-down pattern discovery. The Sourcing-pattern catalog is commencing in Section 6 with a focus on the Sourcing characteristic called *contractual visibility*. After that Section 7 and Section 8 contain pattern specifications belonging to the Sourcing characteristics *monitorability* and *conjoinment*

respectively. In Section 9.4 a case study from the automobile industry is demonstrating how the Sourcing concept and patterns are applied, followed by a discussion in Section 10 that detects the scope for extending the Sourcing concept. Finally, Section 11 gives a conclusion for this paper.

2 Related Work

There are two groups of related work that are discussed in this section. Firstly, several research projects have dealt with the issue of inter-organizational process collaboration in different ways. As this paper tries to explore patterns, the second group of related work presents existing pattern-catalogues for intra-organizational business process perspectives such as control-flow, data-flow, and resource that are a valuable foundation for DIBPM.

With respect to research projects, the WISE project [12, 30] resulted in a software platform for process-based B2B electronic commerce that focusses on support for a network of small and medium sized enterprises. WISE relies on a central workflow engine to control inter-organizational processes that are termed virtual business processes. In WISE a virtual business process consists of a number of black-box services that are linked in a workflow process[12]. A service is offered by an involved organization and can be a business process that is controlled by a local workflow management system.

In the WISE project, local workflow systems are connected as black boxes by a central application, which means no process matching takes place. However, to achieve short production time in B2B, it is a common practice that a service consumer only considers a service provider as a credible supplier when it is possible to perform process mirroring. Thus, although the WISE project succeeds in orchestrating workflows of different collaborating organizations, it does not cater for a flexible degree of mutual visibility of business-process details. Furthermore, collaborating parties can not negotiate how much may be observed during the enactment phase.

In the CrossFlow project [42] inter-organizational business process collaboration was investigated. In the context of this project, the formation of virtual enterprises is realized by dynamic outsourcing. A service matchmaker matches a service offering and a service request. Based on the electronic contract a service enactment infrastructure [24] is established dynamically, based on workflow technology. CrossFlow has an external level that spans across organizational domains where the process specification is part of a contract specification. the workflow specification language of the workflow management system IBM MQSeries Workflow [2] forms the internal process level.

In the CrossFlow project, a part of the service consumer's process are outsourced to a service provider. The latter party has adjustment flexibility of service provision as nodes of the assigned process can be internally refined on a lower process level. However, the service provider should also be able to insert tasks in the provided process that are not merely an internal refinement, but that extend the business process on the highest level. The flexible insertion of internal back-office tasks that remain opaque to the service consumer are one reason why this type of refinement is desirable for a service provider. At the same time it must be ensured that a refinement by inserted tasks

does not violate the service provision behavior a consumer demands. Thus, Sourcing should offer a rigorous foundation to ensure such refinement flexibility.

More recently in the CrossWork [1] project, the objective is to develop automated mechanisms for allowing dynamic workflow formation and enactment, enabling hard collaboration and strong synergies between different organizations. During the course of the project the patterns presented in this paper are incorporated in an XML based language for formulating inter-organizational business process collaboration.

Referencing patterns, Gamma et al. [20] first catalogued systematically some 23 design patterns that describe the smallest recurring interactions in object-oriented systems. Those patterns are formulated in a uniform specification template and grouped into categories. For the domain of intra-organizational business process collaboration patterns were discovered in various perspectives.

In the area of control flow, a set of patterns was generated [7–9] by investigating several intra-organizational workflow systems for commonalities. The resulting patterns are grouped into different categories. Basic patterns contain a sequence, basic splits and joins, and an exclusive split of parallel branches and their simple merge. Further patterns are grouped into the categories advanced branching and synchronization, structural patterns, patterns involving multiple instances, state-based patterns, and cancellation patterns. The resulting pattern catalog is for the evaluation [5, 49] of WSCLs.

Following a similar approach as in the control-flow perspective, data-flow patterns [44] are grouped into various characteristics categories. One category is focuses on different visibility levels of data elements by various components of a workflow system. The category called data interaction focusses on the way in which data is communicated between active elements within a workflow. Next, data-transfer patterns focus on the way data elements are transferred between workflow components and additionally describe mechanisms for passing data elements across the interfaces of workflow components. Patterns for data-based routing deal with the way data elements can influence the control-flow perspective.

Patterns for the resource perspective [43] are aligned to a the lifecycle of a work item. A work item is created and either offered to a single or multiple resources. Alternatively a work item can be allocated to a single resource before it is started. Once a work item is started it can be temporarily suspended by a system or it may fail. Eventually a work item completes. The transitions between those life-cycle stages of a work item either involve a workflow system or a resource. Characteristic categories for the resource perspective are deducted from those life-cycle transitions and group specified patterns.

More recently so-called service interaction patterns [16] are specified for the coordination of collaborating processes that are distributed in different, combined web services. Again, the patterns are categorized according to several dimensions. Based on the number of parties involved, an exchange between services is either bilateral or multilateral. The interaction between services is either of the nature single or multi transmission. Finally, if the bilateral interaction between services is of the nature two ways, a round-trip interaction means the receiver of a response must be equal to the sender. Alternatively a routed interaction takes place.

After related work, the next section presents a specific concept for DIBPM that is the foundation for the pattern analysis of this paper, namely the concept of Sourcing [37]. In the ongoing research project called CrossWork, Sourcing is used as an integral concept.

3 The Concept of Sourcing

Since Sourcing is tackling the issue of structurally matching inter-organizational business processes in the framework of DIBPM, a model is required to manage the complexity resulting from matching and subsequently enacting inter-organizational processes. A definition of DIBPM [21] is given as follows:

A dynamic inter-organizational business process is formed dynamically by the (automatic) integration of the subprocesses of the involved organizations. Here dynamically means that during process enactment collaborator organizations are found by searching business process market places and the subprocesses are integrated with the running process.

Note that at least one organization involved in DIBPM must expose explicit control-flow structure of its business process. Related issues to DIBPM are the definition and identification of processes, the way compatible business partners find each other efficiently, the dynamic establishment of inter-organizational processes, and the setup and coupling for process enactment. In order to manage such complex issues, a three-level framework [23] is a suitable model, which is explained in the next section.

Sourcing within the framework of DIBPM is essential for the automatic structural matching of parties who wish to collaborate. The following definition of Sourcing is used:

In the context of DIBPM, Sourcing is a framework for harmonizing on an external level the intra-organizational business processes of a service consuming and one or many service providing organizations into a B2B supply-chain collaboration. Important elements of Sourcing are the support of different visibility levels of corporate process details for the collaborating counterpart and flexible mechanisms for service monitoring and information exchange.

The definition of Sourcing presented above establishes a differentiation to the research projects mentioned in Section 2 and takes into account requirements stemming from surveying the way industrial partners collaborate in the ongoing CrossWork [1] project.

The Sourcing definition includes the requirement of flexible service monitoring. It is usually not economical or desirable for a consumer to monitor every step of service provision. Likewise, the service provider might not want all enactment progress to be monitored. The WISE project provides tools for evaluating the status of any process for monitoring to allow users to keep track and troubleshoot. Furthermore, WISE includes an awareness model [12] that allows the engine to make informed decisions about configuration, quality of service, resource reservation, load balancing, and so on.

In the CrossFlow project enactment shadowing is performed and explicit quality of service monitoring is taking place. Shadowing means the shared view on the outsourced process is equally reflected in the consumer workflow. Choosing what to monitor from consumer perspective and what to allow to monitor from a provider perspective is negotiated and afterwards specified in an e-contract. However, it is not possible in CrossFlow to flexibly choose different mechanisms of service monitoring, e.g., based on messaging or polling mechanisms.

A crucial issue for collaborating business processes is the overall correct termination. Business processes that terminate correctly in isolation may, for example, contain a deadlock when linked together. In B2B, if an inter-organizational business process collaboration fails, the consequence is penalty payments as a service is not provided as demanded. Such termination problems may also be caused by data-flow through an inter-organizational process. Neither the WISE nor the CrossFlow project offer the option of verifying correct termination before enactment of service provision. On the other hand, in the CrossWork project where Sourcing is used, inter-organizational collaboration needs to be put on a sound, formal foundation. In Sourcing it is the intention to evaluate data flow in an organization, which leads to the generation of control-flow for modelling a process. Before enactment this control-flow is verifiable for correct termination by using the tool Woflan [46]. Consequently, the data-flow perspective can be modelled in this initially created process so that it adheres to the correctly terminating control-flow. That way data-flow problems are reduced substantially. Thus, Sourcing needs to offer explicit control-flow constructs dedicated to data-flow across organizational domains.

The requirements above stress the importance of a sound control-flow foundation for the concept of Sourcing. Thus, the following section informally presents important control-flow properties so that a correct termination of a Sourcing configuration can subsequently be verified with tool support before enactment.

4 Control-flow Basis of Sourcing

The next subsection contains a Sourcing-configuration example based on which important control-flow properties for Sourcing are presented in the following subsection. However, a formal definition of the control-flow properties contained in the following sections is out of scope for this paper.

4.1 An Example

The example of Figure 1 is modelled using labelled Petri nets [40,41]. Squares are denoted as active nodes as they propel the net and circles are denoted as passive nodes representing states. Furthermore, active nodes are always equipped with labels. In a Petri net, passive nodes may contain tokens and active nodes consume these tokens from states. Active nodes are enabled when all their passive input nodes contain at least one token. Such an enabled, active node consumes one token from each of its passive input nodes during firing. After firing, an active node produces tokens that are placed into all its passive output nodes. Arcs in a Petri net do not link two active nodes or two passive

nodes with each other. The special type of Petri nets used in Sourcing, namely WF-nets, has one unique passive input node and one unique passive output node. Furthermore, all other active and passive nodes in a WF-net contribute to its processing.

The depicted Sourcing processes in Figure 1 are distributed across three levels [23]. The very top and bottom show the internal levels of the service consumer and provider where processes are directly enactable by process management applications, e.g., by workflow management systems. Using internal levels caters towards a heterogeneous system environment. In the middle of the service provider and consumer domains, processes are designed in a conceptual level independent from infrastructure and collaboration specifics.

In the center of Figure 1 the external level is stretching across the respective domains of Sourcing parties where process matching takes place. Parts of the respective conceptual-level processes are projected to the external level for performing matching to realize automated and dynamically forged collaboration between partners. A matching is successful when the projected processes of a service consumer and a service provider are equal when similar nodes are embedded in identical control-flow constructs.

Furthermore, Figure 1 shows relevant parts of a Sourcing configuration. Starting with the domain of the service consumer in Figure 1, an in-house process is depicted on the conceptual level that is a WF-net. The in-house process contains a subnet that is termed a consumer sphere that is visualized with a grey ellipse. On the border of the consumer sphere labelled passive nodes are called interface places. Only one interface place is *i*-labelled and only one is *o*-labelled. The other interface places are either *in* or *out*-labelled to represent that exchange direction between the in-house process and its contained consumer sphere. Furthermore, the labelling implies whether an interface place has an input arc or an output arc in the sphere. If an interface place is *i* or *in*-labelled, it has one output arc to an active node in the sphere. If an interface place is *o* or *out*-labelled, it has one input arc from an active node in the spheres.

The in-house process is mapped to the internal level of Figure 1 where legacy systems are located. The consumer sphere is enacted by a different party and therefore projected to the external level to become the consumer's contractual sphere. From the opposite Sourcing domain a complementary provider's contractual sphere is projected to the external level. Since the respective contractual spheres in Figure 1 are equal, consensus is given between the Sourcing parties, which is the prerequisite for a contract. Note that this paper focusses on control-flow and abstracts from other relevant concepts [14] of electronic contracting.

Finally, the Sourcing configuration of Figure 1 includes one service consumer and one service provider. However, Sourcing may be extended for multi-party contracting where many providers offer their services to one consumer, which is demonstrated in the case study of Section 9.4. Such multi-party Sourcing is realized by nesting consumer spheres. Thus, a global consumer sphere contains nested consumer spheres that are embedded in control flow. The nested consumer spheres are filled with refined spheres from service providers.

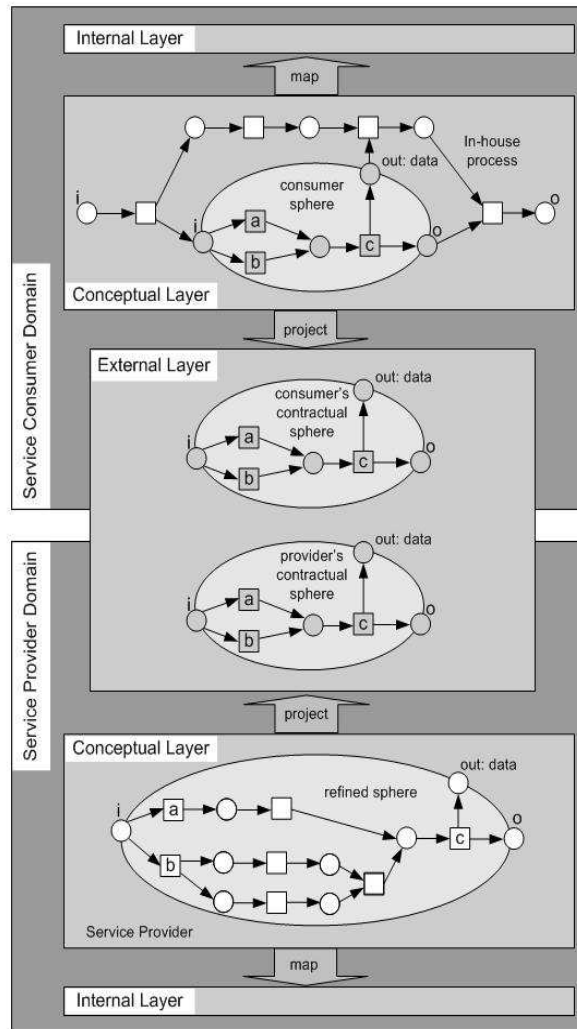


Fig. 1. A three-level business process framework

4.2 Control-Flow Properties of Sourcing

The provider's contractual sphere is complemented by a refined sphere on the conceptual level of the service provider. Compared to the provider's contractual sphere additional nodes are contained on the refined sphere. In Figure 1 such refinement is depicted by unlabelled active nodes in the refined sphere that do not exist in the provider's contractual sphere. Such refinement of the service provider's conceptual-level process must be in accordance with *projection inheritance* [3]. That way the service consumer has assurance the refined process does not violate the desired service-provision behavior during enactment. Projection inheritance is informally defined as follows:

If it is not possible to distinguish the behaviors of processes x and y when arbitrary active nodes of x are executed, but when only the effects of active nodes that are also present in y are considered, then x is a subclass of y .

Thus, process x inherits the projection of the process definition y while process x conforms to the dynamic behavior of its superclass by *hiding* active nodes new in x . Furthermore, processes in an inheritance relation always have the same termination options. For example, in Figure 1 the refining nodes in the provider's conceptual level are visualized as unlabelled nodes that represent, for example, back office tasks. During enactment of the Sourcing configuration in Figure 1, the service consumer only perceives the behavior of the process defined on the external level without having awareness of the refining nodes.

To establish connectability between the consumer sphere, the respective contractual spheres, and the refined sphere, the obligatory requirement of *well-directedness* of a Sourcing configuration must be mentioned. This requirement focusses on the interface places of the spheres, which are part of what can be considered exchange channels between spheres and the remaining in-house process. A Sourcing configuration is well-directed when the interface places of the consumer sphere, the respective contractual spheres of the service consumer and provider, and the refined sphere are equal in number and labelling.

5 Dimensions and Values of Sourcing

Previous sections show the need for a comprehensive concept that tackles structural matching in DIBPM. Consequently, the concept of Sourcing is proposed and defined. In a next step this section pursues a further analysis of Sourcing dimensions and values. In an analogy to earlier analysis of control-flow, data-flow, and the resource perspective, a pattern based exploration of Sourcing is proposed. In the case of the control-flow perspective, a pattern-based analysis [9, 26] has resulted in the development of XML-based languages like XRL and YAWL [33, 38] that have strong control-flow expressiveness. Thus, by pursuing a deeper understanding of Sourcing with a pattern-based analysis approach, an important step is taken to make the concept of Sourcing operational.

5.1 Patterns in Sourcing

By employing the perspective of Sourcing, business processes bridge organizational domains for realizing B2B collaboration. Developers of such Sourcing configurations must understand the concept well. Such developers are likely to be repeatedly confronted with particular questions. How should a problem be solved and what are the pros and cons of a particular solution? Which solution should be chosen and how can the solution be realized? How does the selected solution relate to other potential solutions? Since such questions occur repetitively, developers should not have to consistently reinvent the solutions. Instead, developers of Sourcing configurations should have a set of patterns available that offer standard solutions to problems that keep reoccurring.

Patterns for the control-flow, data-flow, and resource perspective were discovered empirically after evaluating several workflow management systems and WSCLs. However, as the perspective of Sourcing is novel and no supporting systems exist yet, a special approach needs to be taken for discovering patterns. Thus, a particular pattern definition is proposed in line with Section 3 where the Sourcing definition is contained:

A pattern for Sourcing represents a top-down discovered technology independent structure that exists across multiple applications and that has the purpose of offering a predefined, conceptual solution to recurring problems which inter-organizational business process developers are confronted with.

The definition above raises the issue what the specifics of this top-down method are for pattern generation. The following subsection covers this issue.

5.2 A Pattern Discovery Method

In order to discover patterns for Sourcing in a structured, top-down way, the following method is chosen. As depicted in Figure 2, several characteristic dimensions of Sourcing exist in the form of axis that create a multi-dimensional, logical space. In line with the definition of Sourcing contained in Section 3 which is based on observations of industry partners in the CrossWork [1] project, the three dimensions of Figure 2 are contractual visibility, monitorability, and conjunctionment.

On every axis further refining dimension values are located. The axis and their contained values serve as a taxonomy for discovering, ordering, and relating a set of patterns to each other. Furthermore, by evaluating the sort and amount of patterns used, it is possible to position a Sourcing configuration in the multi-dimensional, logical space. In Figure 2 the black circles on the walls of the cube stand for pattern instances that create Sourcing configurations. The connecting lines show the positioning of two concrete configurations where both configuration use four patterns belonging to particular axis values.

The first dimension addresses the issue of keeping business secrets by disclosing different amounts of internal process details to the collaborating counterpart. Thus, depicted dimension values are called white box, grey box, and black box for which patterns are specified in Section 6. The monitorability dimension is focusing on linking equivalent nodes of the consumer sphere and the refined sphere so that their enactment is coordinated and it is possible for one party to observe in a flexible way the enactment progress

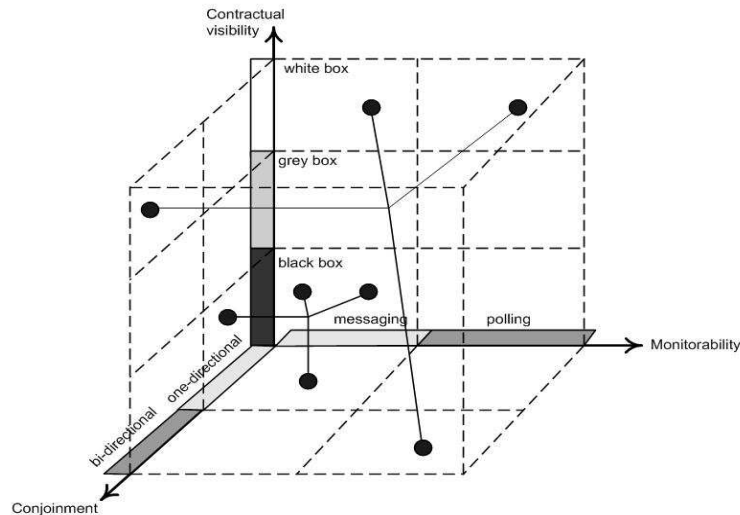


Fig. 2. Dimensions and values of the Sourcing perspective

of the Sourcing counterpart. Given values in Figure 2 are messaging and polling from which patterns are deduced in Section 7. Finally, the conjoinment dimension addresses the issue of modelling the exchange of commercially relevant information between the collaborating domains. Such information exchange is either one-directional or bi-directional and Section 8 specifies patterns that are deduced from the given values.

5.3 A Pattern Specification Template

For specifying discovered Sourcing patterns in the following sections, a description template is used containing a name, problem statement, pattern description, several forces, and one or several examples:

- The *name* is an identifier of a pattern that needs to be meaningful and representative for its main ideas.
- The *problem* of a pattern is a statement describing the context of pattern application. In this context conflicting environmental objectives and their constraints are described. The application of a pattern in that context should result in an alignment of the given objectives.
- The *description* of a pattern mentions the inherent, differing pattern properties and describes the relationship between them.
- The *forces* describe obstacles that occur during pattern application and that may prevent an alignment of objectives of a pattern context.
- Finally, the *example* of a pattern either describes a concrete instance in a real-world setting where the pattern is used or an abstract application scenario. Furthermore, the *intuitiveVisualization*

Next, the following section specifies three patterns in the mentioned pattern-description template that belong to the contractual-visibility dimension.

6 Contractual Visibility

The three-level model of Figure 1 shows two identical contractual spheres contained in the external level of a Sourcing configuration resulting from a consensus between service consumer and provider. The contractual visibility is variable depending on the amount of process content that is projected from the consumer sphere and the refined sphere to the contractual sphere. On the one hand the consumer sphere and the contractual spheres contain identical nodes and control-flow constructs. Such an example is depicted in Figure 1. On the other hand only the interfaces of the consumer sphere are projected to the contractual sphere, which grants the service provider no visibility of further process details in the consumer sphere. Finally, a third option of contractual visibility is located between the extremes where the consumer sphere's interfaces and a subset of the remaining process content is projected to the contractual sphere. Corresponding to the values on the contractual-visibility axis of Figure 2, the emerging patterns for contractual visibility are called *black box*, *grey box*, and *white box*.

The figures below depict contractual-visibility patterns, only contain two levels, namely the service provider's and consumer's conceptual level, and the external level. The reason is that the consumer's and provider's contractual spheres always need to be similar for representing a consensus. Thus, depicting both contractual spheres does not add to the specification of contractual-visibility patterns. Furthermore, in the consumer's contractual level only the sphere is depicted and the rest of the in-house process omitted as the content of the in-house process has no influence on the type of contractual-visibility pattern. Finally, when an active node is depicted with a τ label [4] in a consumer sphere or a refined sphere, it means the Sourcing counterpart is not aware of this active node during the build time and the enactment of a Sourcing configuration. During build time the Sourcing counterpart is not aware because a τ labelled active node is not projected to the external level and during enactment the counterpart is not aware since the effects of enacting a τ labelled active node are hidden from the counterpart.

Pattern 1 (Black Box)

Problem: The service consumer is interested in using service provision. However, the consumer does not mind how the provision is carried out as long as the specified exchanges are performed correctly.

Description: In the case of black-box visibility, only the interfaces of the consumer sphere are focused on. These interface places must be equally contained with identical labels in the consumer sphere and the refined sphere. It is not permitted for any sphere of the external level or the respective conceptual levels to have a deviating set of interface places or differing labels. The opposing Sourcing parties are not aware of other details in the consumer sphere and the refined sphere since the contractual spheres only contain interface places with their labels. Thus, the refined sphere may otherwise completely

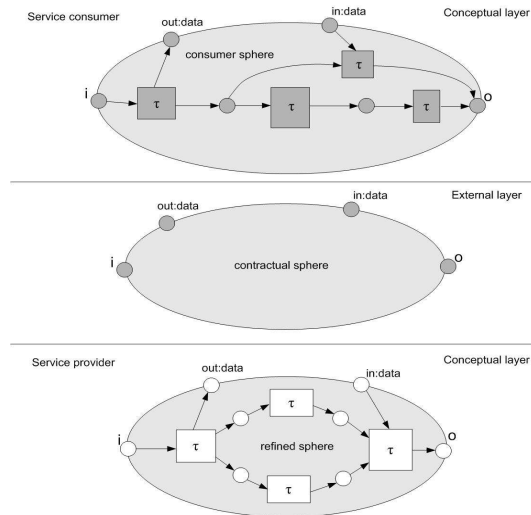


Fig. 3. Black-box pattern example

deviate from the consumer sphere provided the similar interface places are serviced correctly according to their labels.

Forces: When only the interfaces of a contractual sphere are given, the service provider might struggle to fill the refined sphere with process content. During the enactment of a Sourcing configuration the consumer might experience service provision that is counter productive.

Examples:

- A travel agency organizes journeys abroad for arbitrary customers. Such a travel package consists of booking a trip, finding a hotel, renting a car, and so on. The travel agency does not specialize on negotiating and ordering affordable hotel bookings. As a result such a service is sourced in from service brokers where providers are registered who know the market and have special agreements with hotels in proximity. Since the concrete procedures of finding and booking most suitable hotels differs from country to country, the travel agency merely discloses the interfaces for starting and ending the hotel-booking service and additional interfaces for communicating with the provider service the travel details and receiving the billing details.
- In Figure 3 an abstract example of a black-box pattern is presented. The *i* and *o*-labelled input and output places and the *in* and *out*-labelled interface places are fully disclosed in the contractual sphere. Consequently these interface places are also part of the refined sphere. However, it is depicted that all active nodes contained in the consumer sphere and the refined sphere have a τ label, which means the Sourcing counterparts are not aware of the other's process content. Additionally, Figure 3 depicts that control-flow constructs in both conceptual-level spheres are

different. However, these differences do not result in soundness problems as long as the interface places are adhered to. The reason is that the consumer sphere and the refined sphere are WF-nets when the *in* and *out*-labelled interface places are removed.

Pattern 2 (White Box)

Problem: The service consumer demands an externalized service where the provider must strictly adhere to the requirements defined in the consumer sphere. Despite having strict service requirements imposed, the provider should still have the flexibility to adjust his service provision to internal requirements. However, these internal adjustments should remain hidden from the service consumer.

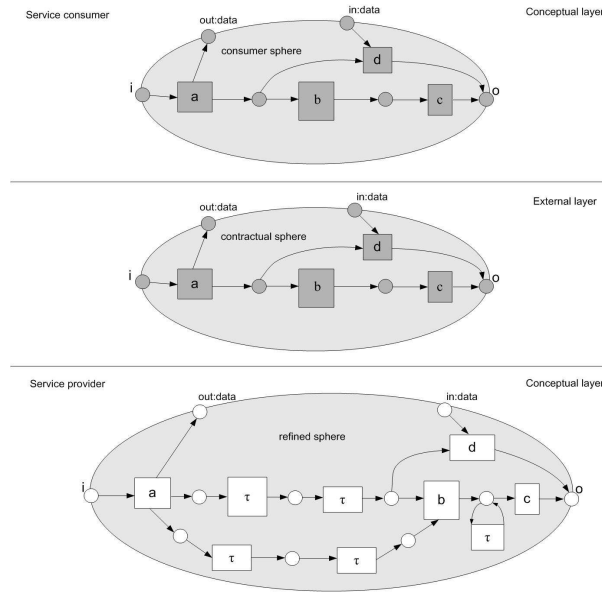


Fig. 4. White-box pattern example

Description: A consumer sphere is fully projected to the external level of a Sourcing configuration. All labels are visible to the provider as they are fully projected to the consumer's contractual sphere. As a result the nodes and labels of the contractual sphere are all present in the refined sphere. However, it is possible for the service provider to insert additional active nodes in the refined sphere in accordance with projection inheritance [3].

Forces: Adhering to projection inheritance in a refined conceptual sphere is not trivial for a service provider as the correct application of refinement rules requires deep knowledge of modelling formalisms when no tool support is available. However, with tool support projection inheritance can be verified by replacing the consumer sphere of

the in-house process with the refined sphere. Projection inheritance is given when the resulting net is a subclass of the in-house process.

Examples:

- For a so-called original equipment manufacturer (OEM) in the automobile industry it is important to reduce the production time per truck to 14 working days. In order to achieve this objective, the OEM pursues a tight integration of suppliers. Given the complexity of producing a truck within the market-dictated time budget, the OEM demands that providers precisely mirror processes that are outsourced.
- Figure 4 depicts that all nodes, control-flow constructs, and labels of the consumer and contractual sphere are similar and no deviations are contained. While the refined sphere at the bottom of Figure 4 contains all elements of the contractual sphere, many additional elements are depicted with active nodes containing a τ label. This means the service consumer is not aware of these additional active nodes during build and run time.

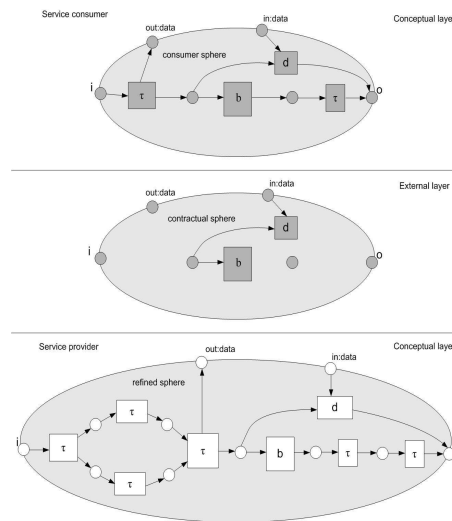


Fig. 5. Grey-box pattern example

Pattern 3 (Grey Box)

Problem: The service consumer does not mind in large parts how service provision is realized. However, the consumer wants to ensure that particular steps contained in the provided service are mandatory and carried out in certain control-flows.

Description: The pattern does not permit deviating interface places in the consumer sphere, the contractual sphere, and the refined sphere. Furthermore, a subset of nodes different from interface places contained in the consumer sphere is projected to the

contractual sphere. As a result many nodes in the consumer sphere are therefore not visible to the service provider. The subset of nodes contained in the contractual sphere must be equally present in the refined sphere. To recomplete the demanded service provision, the provider fills the sphere adopted from the external level with additional nodes that are opaque for the service consumer. The resulting refined sphere is must be a WF-net.

Forces: The service consumer can not demand from the provider adherence to projection inheritance since only parts of the overall consumer sphere are disclosed.

Examples:

- An OEM in the automobile industry demands from a provider the delivery of leather coated car seats. However, the OEM demands from the provider to purchase the leather from a special certified seller following an agreed upon flow of tasks.
- An example of grey-box contractual visibility is depicted in Figure 5. The contractual sphere depicts interface places with labels that are identical compared to the consumer sphere and the refined sphere. The contractual sphere shows two process places and two labelled, active nodes that are equally contained in the consumer sphere and the refined sphere. However, the the latter two spheres fill the gap depicted in the contractual sphere with differing nodes embedded in different control-flow constructs. Since active nodes carry τ labels, the Sourcing counterparts are not aware how the sphere in the opposing domain is completed.

In the next section patterns belonging to values of the dimension called monitorability are presented. These monitorability patterns ensure different levels of enactment awareness for the service consumer.

7 Monitorability

Revisiting the initial Sourcing example of Figure 1, there is no linking depicted between the contractual sphere, refined sphere, and the contractual spheres of the respective Sourcing domains. However, during enactment of a Sourcing configuration, the collaborating parties must be able to coordinate and overview in a flexibel way the service provision and consumption. Thus, the monitorability patterns of this section offer ways of establishing such links between spheres in different domains.

Communication across organizational domains may take place in two ways. Accordingly, in Figure 2 two values are contained in the dimension called monitorability, namely polling and messaging. Based on those generalized communication concepts, patterns for Sourcing are deducted. In the case of value *polling* one Sourcing domain periodically asks if a change has taken place to a linked node of the opposing Sourcing domain. Detected changes are duplicated by the linked node in the polling Sourcing domain. The second monitorability classifier called *messaging* reverses the signalling direction. When a linked node experiences a change, a signal is sent to the other linked node in the opposing Sourcing domain. Investigating such opposing monitorability patterns is relevant to cater for a heterogenous enactment environment that does not cater for similar monitorability options in the opposing Sourcing domains.

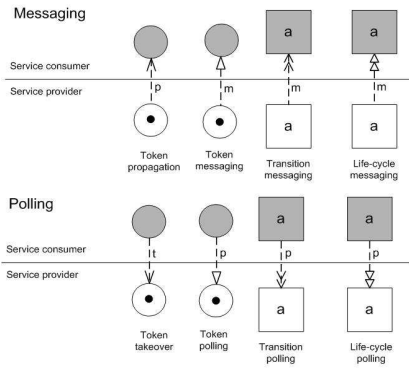


Fig. 6. Linking options for pursuing run-time visibility

The degree of monitorability for a service consumer during the enactment of a Sourcing configuration is determined by the level of contractual visibility (see Section 6). Only nodes from the consumer sphere and the refined sphere of a Sourcing configuration that are projected to the respective contractual spheres on the external level are considered for monitorability. Figure 6 depicts different messaging and polling patterns for linking such passive and active nodes without claiming completeness. However, before messaging-monitorability patterns are specified in the following subsection, an investigation of issues related to the monitorability of equally labelled active nodes with a life cycle is carried out

7.1 Introduction of Life-Cycle Monitorability

For applying patterns of life-cycle monitorability, it is necessary to explore whether it is possible to achieve a mapping of life-cycle stages that equally labelled, active nodes use in opposing Sourcing domains. For presenting an analysis of the problem in this subsection, a life cycle of an active node is refined to a labelled WF-net when an isolated active node has only one passive-input node and one passive-output node. As Figure 7 shows, there is only one life-cycle transition serving as an output node of that active node's passive-input node and only one life-cycle transition serving as an input node of the active node's passive output node. Those life-cycle transitions propel an active node's life-cycle states, which are represented by labelled passive nodes.

If an active node in a process has multiple passive-input nodes, they are at the same time multiple passive-input nodes of the first labelled life-cycle transition that is starting to propel the life cycle. Equally, when the active node with a life cycle has multiple passive-output nodes, they are at the same time the passive-output nodes of the last unique, labelled life-cycle transition.

In Figure 7 the respective active nodes' life cycles are WF-nets as both have one passive input and one passive output node. The active nodes are depicted by grey shaded boxes. All other nodes are connected, i.e., they have at least one input and one output arc. In

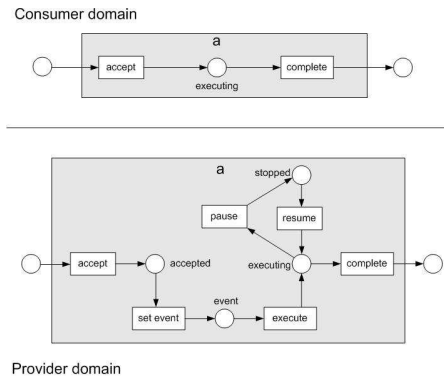


Fig. 7. Mapping of life-cycle stages

both cases the life-cycle nets terminate after enactment so that only one token is left in their passive-output nodes. Furthermore, Figure 7 contains a loop for pausing and resuming an active node that is in the life-cycle stage *executing*. The active node contains an additional life-cycle transition and state for event setting, which is not present in the consumer's life cycle. When an active node is accepted by a provider, this may serve as an event that other active nodes might need to wait for before they can be enabled.

Revisiting the notion of projection inheritance in Section 3, the life-cycle of the provider's isolated active node in Figure 7 is a subclass of the consumer's active-node life cycle. Thus, in this case, when all active nodes in a refined sphere and an in-house process are replaced with life-cycle transitions and life-cycle states then an in-house process containing a refined sphere instead of the consumer sphere is still a sound WF-net. Next, the patterns depicted in Figure 6 are described in further detail starting with patterns for the monitorability value called *messaging*. For all specified monitorability patterns the examples are of an abstract nature and adjusted to purposeful applications in a Sourcing configuration.

Pattern 4 (Token Propagation)

Problem: During the enactment of a Sourcing configuration, tokens enter *out*-labelled interface places in the refined sphere that should trigger a message exchange with the in-house process. Since all passive nodes of the refined sphere must be empty after enactment, these tokens should be removed while the exchange between the refined sphere and the in-house process takes place. Additionally, the final token left in the *o*-labelled output place of the refined sphere needs to be removed to complete enactment.

Description: This pattern links two equally labelled passive nodes from spheres that are not part of the same level in a Sourcing configuration. Thus, linked are *out* and *o*-labelled interface places of the refined sphere, contractual sphere, and the consumer sphere. The propagation starts when a token arrives in the linked passive source node. In that case the token is passed on as a message to the linked passive target node in the

different level. As a result the passive source node has that token removed and placed in the passive target node.

Forces: A token already resides in the *out*-labelled interface place of the consumer sphere before token propagation takes place. Thus, this token may be ahead of the refined sphere's *out*-labelled interface place and as a result it still takes more time until a token-propagation message is triggered. Such an 'early' token can not result in enabling an active node.

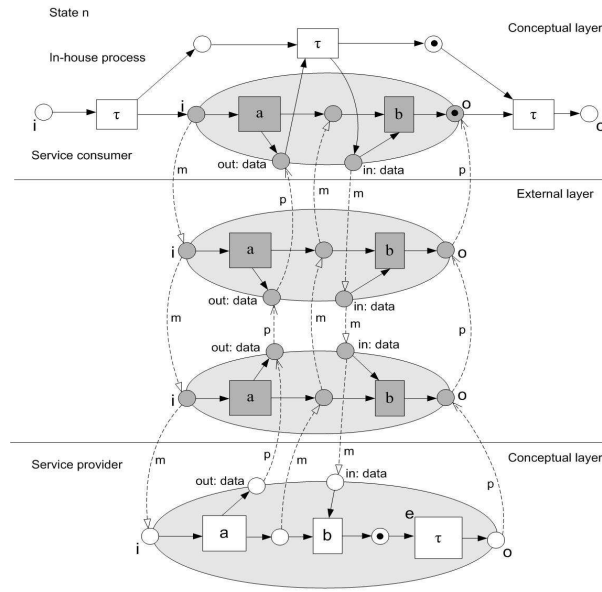


Fig. 8. Example of token propagation and token messaging

Examples:

- For *o*-labelled interface places, the Sourcing configuration depicted in Figure 8 shows an example of an 'early' token occurring in combination with token propagation occurring in combination with token propagation. A small *e* shows which active nodes are enabled. The enactment state of Figure 8 shows a token in the consumer sphere's *o*-labelled interface place that is early. The reason is an additional active node with attached τ label in the refined sphere at the left bottom of Figure 8 that is enabled but hasn't fired yet. The active node of the in-house process that follows the consumer sphere's *o*-labelled interface place can not be enabled as the contained token is too early produced after the firing of the *b*-labelled active node. When the τ -labelled active node of the refined sphere completes firing, a token is placed in the output place. Consequently, the triggered token-propagation message results in having the token removed from the *o*-labelled

interface place of the refined sphere. Next, the early token in the consumer sphere turns 'current' and enables the active output node of the in-house process.

- The *out*-labelled interface place of the refined sphere in Figure 8 is connected with a token-propagation arc. When a token is placed in the interface place, a message is sent to the equally labelled interface place of the consumer sphere. As a result the token is passed on across the Sourcing domains from one interface place to the other.

Pattern 5 (Token Messaging)

Problem: During the enactment of service provision, the consumer wants to observe certain provision states. Thus, state changes that occur in the refined sphere of the provider domain should be mirrored by the consumer sphere in the opposing domain.

Description: The token-messaging mechanism is triggered when a passive source node experiences a change in the contained amount of tokens. A message delivers the new number of contained tokens to the passive target node. The passive target node evaluates if its contained token amount deviates from the number delivered in the received message. If the number deviates, the tokens contained in the passive target node are synchronized while the amount of tokens in the passive source node remain unchanged.

Examples:

- Figure 8 depicts a token-messaging example for *i*-labelled interface places. The token-messaging direction must lead from the consumer sphere to the refined sphere. The reason for this linking direction is that an active input node belonging to the service consumer's in-house process puts a token into the consumer sphere. Consequently, the enactment of the consumer sphere is started. In order to start the enactment of the refined sphere, token messaging is required to put a token into the *i*-labelled interface place of the refined sphere. After token messaging, the input places of both spheres contain a token and the enactment of the Sourcing configuration may carry on.
- To support monitorability for the service consumer, passive nodes in spheres that are not interface places can be linked in the direction from the provider domain to the consumer domain. Once the enactment of service provision is started, state changes are taking place in the refined sphere that should be monitorable in the consumer sphere. Thus, token messaging from the provider domain to the consumer results in having state changes of service provision followed for permitting consumer monitoring. In Figure 8 corresponding examples are depicted.
- Interface places with an *in* label have an active input node from the in-house process outside of the consumer sphere. As Figure 8 shows, such a token enables the active receive nodes in the consumer sphere and the refined sphere. Therefore, token messaging is applicable in such a case with a linking direction from the consumer sphere to the refined sphere.

Pattern 6 (Transition Messaging)

Problem: An active node does not contain a lower-level life cycle, i.e., it is a transition. An equivalently labelled transition in the consumer domain only has to be enacted when the linked transition in the provider's refined sphere has fired.

Description: While a linked source transition fires, a message is sent to the equally labelled target transition. If the target transition is enabled, i.e., has a token in all its input places, the transition fires once the message from the equally labelled source transition arrives. Consequently, the target transition produces a token for all its output places.

Examples:

- An example of transition messaging is depicted in Figure 9 in connection with active nodes that have life-cycles. When enabled, the consumer's life-cycle transitions can fire in synch with an equally labelled life-cycle transition of a service provider. The life-cycle transitions labelled *accept* and *complete* equally occur in active nodes of the service consumer and provider domain. Thus, these two nodes are suitable for linking with transition-messaging arcs directed from the service provider to the consumer.
- Active nodes without a life cycle can be linked with a transition-messaging arc. These nodes must have equal labels and be mutually known, i.e., be part of the respective contractual spheres on the external level of a Sourcing configuration. Then the linking direction is from the domain of the service provider to the domain of the consumer.

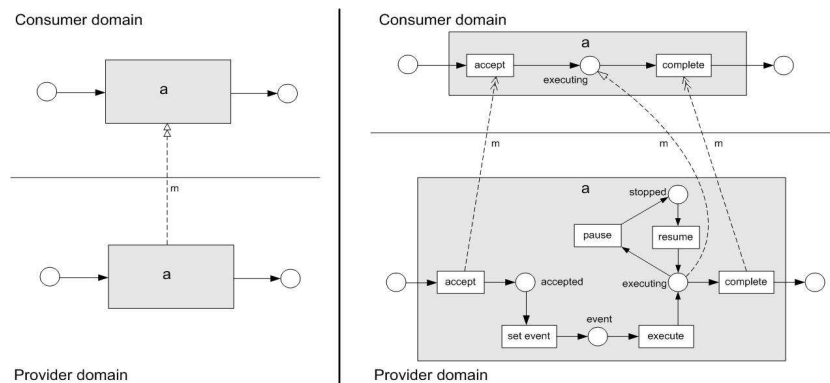


Fig. 9. Life-cycle messaging as a black box and white box

Pattern 7 (Life-Cycle Messaging)

Problem: During the enactment of active nodes carrying labels that are equally located in the domains of the respective Sourcing parties, the life-cycle changes need to be communicated from the domain of the service provider to the consumer's domain.

Description: This pattern assumes the life-cycle of active nodes that are linked for monitoring consist of lower-level nets. Consequently, life-cycle states and life-cycle transitions that are semantically matched for the respective active nodes, are linked with token messaging and transition messaging arcs.

Forces: Prior semantic life-cycle mapping is required in the heterogeneous system environments of the service consumer and provider. As discussed in the beginning of Subsection 7.1, linked active nodes might have different life cycles with steps not present in the opposing Sourcing domain or with differently positioned life-cycle steps. Some life-cycle steps may be mutually present but differently termed. Thus, a decision needs to be taken about the semantic equivalence of particular life-cycle steps. Equivalently to the depiction of Figure 8, life-cycle messaging is also confronted with the problem of 'early' tokens.

Examples:

- In the example depicted in Figure 9, the life-cycle states in the consumer's and provider's active node carry the equal *executing* label. Thus, the respective life-cycle states are linked with a token-messaging arc (see Pattern 5). During the enactment of a Sourcing configuration the life-cycle transitions in the refined sphere propel the life cycle of active nodes. Tokens that appear in the source life-cycle state are messaged to the equally labelled target life-cycle state in the consumer domain.
- Equally labelled life-cycle transitions are linked with a transition-messaging arc (see Pattern 6) from the direction of a service provider to a consumer. An example is given in the previous pattern.

The following patterns are focusing on different ways of *polling* changes during the enactment of a Sourcing configuration. Those polled changes help one Sourcing party to monitor and mirror the enactment progress of the opposing Sourcing party. In Figure 6 the monitorability patterns that use polling mechanisms are depicted at the top. These patterns are described below in further detail:

Pattern 8 (Token Takeover)

Problem: A refined sphere needs to be cleared of tokens residing in passive nodes that do not serve as input nodes to any other active nodes. However, the token clearing is triggered from the domain of the service consumer.

Description: Either *o*-labelled or *out*-labelled interface places are linked in the direction from a consumer sphere to the corresponding refined sphere of a provider. A request is sent periodically from the domain of a service consumer to the equally labelled passive node of the refined sphere to check whether a token resides there. If the response is positive, the token is removed from the provider domain and placed in the target interface place of the consumer domain.

Forces: When *o*-labelled interface places of the service consumer and provider are linked by using a token-takeover pattern, a special situation occurs with respect to an 'early' token that arrives in the consumer's *o*-labelled interface place. A request is triggered for token takeover from the domain of the service consumer. If the reply message from the provider domain states no token is contained then active output nodes of the consumer sphere's *o*-labelled interface place are not enabled by that 'early' token.

Examples:

- The *out*-labelled interface place of the refined sphere in Figure 10 is connected to a token-takeover arc. When a token is placed in that interface place, token polling

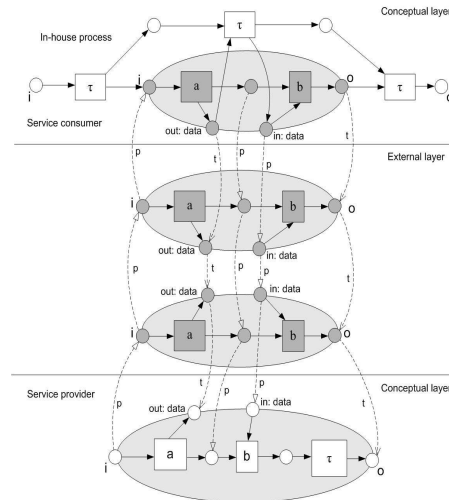


Fig. 10. Example of token takeover and token polling

from the domain of the service consumer results in a positive response. As a result the token is taken over and placed in the equally labelled interface place of the consumer's domain. Consequently, the provider's *out*-labelled interface place is empty.

- For *o*-labelled interface places, the Sourcing configuration in Figure 10 shows an application of token takeover. Periodically the provider's interface place is polled for contained tokens that need to be removed and placed in the *o*-labelled interface place of the consumer. When a token is contained in the latter interface place, the following τ -labelled output node that belongs to the in-house process, is not enabled. Instead a token needs to be taken over first from the service provider's *o*-labelled interface place to make the 'early' token in the consumer's sphere 'current'. As a result the τ -labelled active-output node is enabled and ready to fire.

Pattern 9 (Token Polling)

Problem: During the enactment of a Sourcing configuration the opposing Sourcing parties want to monitor the progress of state changes. However, for economic reasons only some state changes are of relevance to a Sourcing party and are consequently mirrored according to the opposing domain.

Description: Passive nodes that belong to the domains of a service provider and consumer and that are not *out* and *o*-labelled interface places are linked with each other. Periodically the number of tokens contained in the respective passive nodes are checked. When the numbers deviate, the amount of tokens contained in the passive node of the polling domain is synchronized if it is less than in the target domain. Such synchronization does not affect the token amount contained in the passive node of the target domain.

Forces: The proper setting of the polling interval is relevant for keeping the polling domain up to date with respect to state changes in the target domain. If the polling intervals are too long, state changes of the target domain can be missed. On the other hand, if the polling intervals are very short, the performance of the applications involved in the enactment of a Sourcing configuration are stressed unnecessarily. Furthermore, it must be stated that early tokens may occur in a linked passive node of a consumer sphere. Thus, such an early token only enables the output transitions of the place it resides in when the procedure of token polling results in a synchronization of the amount of tokens compared to the linked passive node in the refined sphere.

Examples:

- The *i*-labelled interface place of the consumer sphere and refined sphere in Figure 10 are linked with token-polling arcs in the direction from service provider to consumer. The reason for the linking direction is that an enactment-initializing token reaching the *i*-labelled interface place of a consumer sphere is at the same time the trigger for starting the enactment of the refined sphere.
- The *in*-labelled interface places of Figure 10 are linked with a token-polling arc from the domain of the service consumer to the provider. The token is produced by an active input node belonging to the consumer's in-house process and consumed by an active receive node that is located in the refined sphere.
- Passive nodes that are not interface places are linked with token-polling arcs in the direction from service consumer to the provider domain. In Figure 10 the passive node between the *a* and *b*-labelled active nodes of the refined sphere is polled for tokens. The linked place in the consumer sphere synchronizes its token number.

Pattern 10 (Transition Polling)

Problem: A service consumer wants to monitor when an enabled active node without a life cycle, i.e., a transition, may fire in its domain.

Description: Two transition nodes with equal labels in opposing Sourcing domains are linked in the direction from service consumer to provider. When the consumer's transition is enabled, periodic polling takes place to check if the provider's linked transition has fired. If the response is positive, the consumer transition fires and the polling is stopped.

Forces: The forces exerted on this pattern are comparable to the forces of Pattern 9 during the enactment of a Sourcing configuration with respect to the alignment promptness of a source transition compared to the polled target transition.

Examples:

- An example of transition polling is depicted in Figure 11 in connection with active nodes that have life-cycles. Both life-cycle transitions labelled *accept* and *complete* occur in active nodes of the service consumer and provider domain. The equally labelled life-cycle transition of the service provider is periodically polled. If the response contains the message that firing has been carried out, the consumer transition fires, which results in a change of the life-cycle state.
- Transitions, i.e., active nodes without life cycles, that are equally labelled in the domains of the service consumer and provider may be linked with transition-polling arcs. When the consumer's transition is enabled, polling of the target transition in

the provider's domain starts to monitor its firing. This is periodically repeated until the response states firing occurred. As a result the consumer transition also fires to follow the provider domain and polling is stopped.

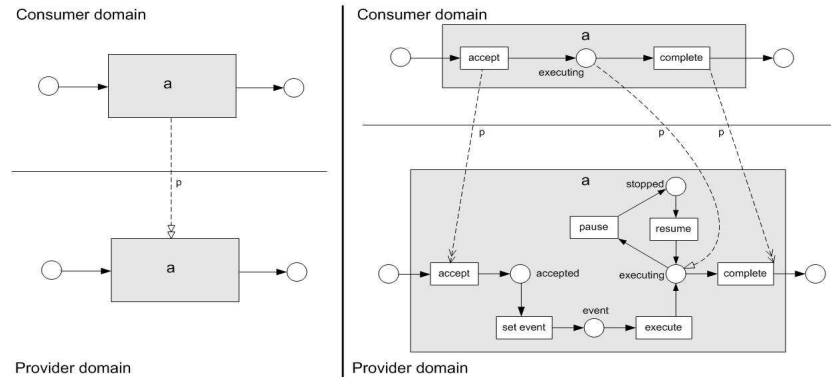


Fig. 11. Life-cycle polling as a black box and white box

Pattern 11 (Life-Cycle Polling)

Problem: During enactment of a Sourcing configuration the life cycles of active nodes in the consumer sphere need to be synchronized with the life-cycles of equally labelled active nodes in the provider's domain. However, the initiative for triggering alignments during service enactment should occur in the consumer domain.

Description: This pattern assumes the life-cycles of active nodes consists of a lower-level net (see Subsection 7.1). It is a prerequisite that life-cycle states and life-cycle transitions need to be semantically matched across Sourcing domains. Consequently they are linked in the direction from the service consumer to the provider domain with token-polling and transition-polling arcs.

Forces: For this pattern the forces are comparable to those mentioned in Pattern 7 about life-cycle messaging. Additionally, the polling intervals must be set appropriately. The consumer sphere runs the danger of falling behind during enactment when polling is not performed promptly. However, too frequent polling is a burden for the efficiency of applications that are used during the enactment of a Sourcing configuration.

Examples:

- In the example depicted in Figure 11 the life-cycle states labelled *executing* in the consumer's and provider's active node are linked with a token-polling arc (see Pattern 9). During enactment of a Sourcing configuration the provider's active node is polled for having reached the life-cycle stage *executing*. If the response results in having a lower number of tokens contained in the respective life-cycle state, the number of tokens is aligned in the domain of the service consumer. Consequently, the tokens in the provider's life-cycle state remain unchanged.

- Equally labelled life-cycle transitions are linked with a transition-polling arc from the direction of a service consumer to a provider domain. An example is given in Pattern 10.

Next, the Sourcing-patterns space in Figure 2 contains another Sourcing dimension that is covered in following section, namely conjunctionment.

8 Conjunctionment

Conjunctionment is focusing on the way a service provider and consumer establish exchange channels with each other. In the example of Figure 1, the refined sphere, respective contractual spheres, and the consumer sphere contain *out*-labelled interface places through which such conjunctionment between the two Sourcing domains takes place.

In Figure 12 an extra notation for active nodes is presented that is used for further discussing the conjunctionment dimension. By using this notation, particular features of conjunctionment patterns become apparent. The active nodes of Figure 12 either receive a message, send a message, or receive and send a message consecutively. Such nodes can either be transitions that fire immediately when enabled or they contain more elaborate life cycles (see Subsection 7.1).

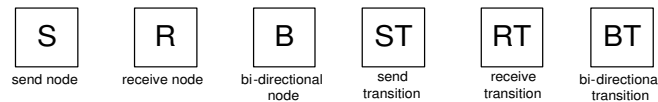


Fig. 12. Active conjunctionment node notation

In order to make those distinctions visible, additional labels are depicted in Figure 12 that are carried by the shown nodes. The labels *S*, *R*, *B* are for active nodes with lower level life cycles where the first is sending or initiating an exchange, the second receives or accepts an exchange, and the latter is bidirectional, i.e., the node needs to accept an exchange before it is enabled to initiate a counter exchange after firing. The labels *ST*, *RT*, *BT* are for active nodes that are transitions without any contained life cycle where the first is for an exchange-initiating transition, the second an exchange-accepting transition, and the latter node is a bi-directional transition that needs to accept an exchange before an exchange is initiated in response.

In accordance with the values depicted in Figure 2 on the conjunctionment dimension, the first two patterns described below are part of the conjunctionment value *one-directional* and the latter two patterns are part of the value *bi-directional*. The reason for having two patterns deducted from each conjunctionment value is that an information exchange can either be initiated from the domain of the service consumer or from the domain of the service provider.

Every pattern specification includes a visualization that consists of two parts. The top shows the conceptual-level in the consumer domain consisting of an in-house process with a contained consumer sphere. Due to the requirement of well-directedness (see Section 3) the conjunction pattern of the top figures is replicated in the contractual spheres and the refined sphere. Thus, for sake of brevity a depiction of the external level and the provider's conceptual level are omitted. If the labels of conjunction nodes in the top depictions are in brackets, then the service provider is not aware of them as they are part of the in-house process. The same is true for active nodes that carry a τ -label. The bottom of the pattern figures show pattern variations that employing different types of conjunction nodes from Figure 12. The specified conjunction patterns do not contain a description of related forces as they are similar. Instead the forces are given towards the end of this section.

Pattern 12 (Provider-Initiated One-Directional)

Problem: During build time of a Sourcing configuration a construct is required that allows the service consumer and provider to reach consensus on an exchange of information directed from the provider domain to the consumer domain. The created exchange should permit the checking of correct termination.

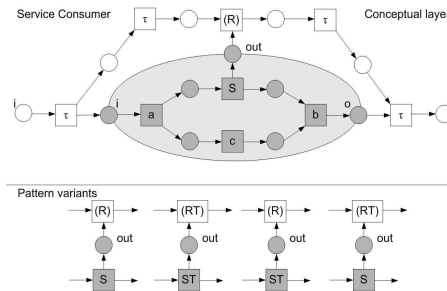


Fig. 13. Provider-initiated one-directional conjunction

Description: All spheres of a Sourcing configuration have a port through which information leaves towards the domain of the service consumer. This port has an input node for triggering the sending of information. In the domain of the service consumer an output node of the port receives the delivered information. This receiving conjunction node is part of an in-house process outside of the consumer sphere.

Examples:

- A service provider in the automobile industry agrees to supply an engine to an original manufacturer (OEM) of trucks. The OEM demands process mirroring for achieving a tight integration with the provider. Whenever the production of an engine is completed, the OEM demands to have the resulting data from the quality checks delivered for evaluation and controlling purposes.

- An abstract example is depicted at the top of Figure 13 where a conceptual-level process in the consumer domain is depicted. The consumer sphere contains a sending conjunction node connected to an *out*-labelled interface place. These two nodes are replicated in all other spheres that are part of the same Sourcing configuration if they are projected to the consumer's contractual sphere. The in-house process of Figure 13 has an *R*-labelled conjunction node that uses the interface place as an input node. Since the enactment of the sending conjunction node takes place in the domain of the service provider, an exchange from provider to consumer takes place via the external level.

Pattern 13 (Consumer-Initiated One-Directional)

Problem: The Sourcing parties need to reach consensus on an information exchange directed from the consumer domain to the provider domain. Such an exchange should not endanger the correct termination of the overall Sourcing configuration.

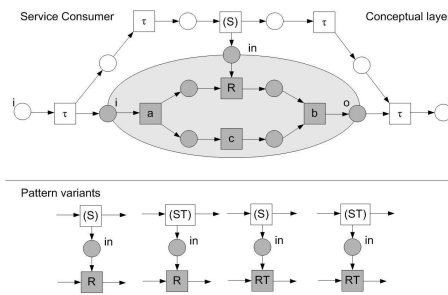


Fig. 14. Consumer-initiated one-directional conjunction

Description: In the domain of the service consumer a sending conjunction node that initiates information exchange to the domain of the service provider, is part of an in-house process outside of the consumer sphere. Such an active node is connected to a port through which information is injected into the spheres of the Sourcing configuration. The input port is followed by a node that receives this information and which is located in the consumer sphere, the respective contractual spheres, and the refined sphere.

Examples:

- In alignment with the first business example given in Pattern 12, the exchange is now directed from the OEM to the service provider who delivers engines for truck production. The produced trucks may individually vary depending on customer demand. Thus, for every engine that is produced by the provider, the adjusted engine specifications must be communicated as input to the provider. That way the overall Sourcing configuration is stable from a process point-of view. However, it is possible to adjust the engine production in a flexible way to requested engine variations.
- The consumer's in-house process is depicted in Figure 14 where it has a sending conjunction node equipped with an *S* label in brackets and a passive output node

that is an *in*-labelled interface place. This interface place in the domain of the service consumer serves as a passive input node for a receiving conjunction node contained in the consumer sphere. In Figure 14 that latter node carries an *R* label. Due to well-directedness, the *in*-labelled interface place is replicated and contained in the refined sphere of the service provider. During enactment of the overall Sourcing configuration an exchange is carried out that is initiated by the consumer and accepted by the provider. If data flows along such control flow that is verified for soundness, it is likely that the exchange from the domain of the service consumer to the provider does not create problems.

Pattern 14 (Provider-Initiated Bi-Directional)

Problem: The service provider has to forward information to the consumer domain that should immediately be answered by a response. Such an exchange should not violate the correct termination of a Sourcing configuration.

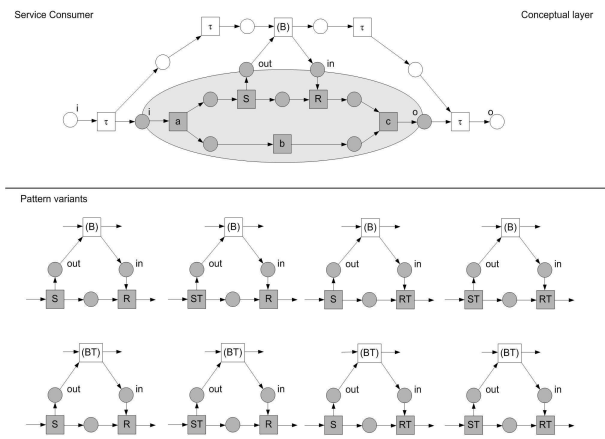


Fig. 15. Provider-initiated bi-directional conjunction

Description: A sending conjunction node that initiates information exchange to the domain of the service consumer is part of the consumer sphere, the contractual spheres, and the consumer. Such a sending conjunction node is connected to a port at the border of the sphere through which information is exchanged to the domain of the service consumer. The port is connected to a bi-directional conjunction node that is part of an in-house process outside of the consumer sphere. This bi-directional node receives the delivered information and responds with sending information back to the domain of the service provider through another sphere port. Finally, a receiving conjunction node that is replicated in all spheres of a Sourcing configuration serves as a recipient for the information sent through the response port.

Examples:

- A local service provider is booking a hotel room according to special requirements of a customer. Those requirements are delivered by a travel agency that sources in the service of booking a hotel room. Since the payment format should be kept flexible, the provider must be able to ask the travel agency during enactment how the customer would like to pay for a particular hotel room. The response from the travel agency should be received immediately by the service provider in order to finalize the hotel booking.
- In the top of Figure 15 the depicted consumer sphere contains a sending and receiving conjunction node in the sphere. The first node carries an *S* label and the latter an *R* label. The sending conjunction node has a passive output node that is an *out*-labelled interface place, which serves as an input node for the bi-directional node in the in-house process. This *B*-labelled conjunction node produces an immediate response during the exchange. For this reason one passive output node of the bi-directional conjunction node is an *in*-labelled interface place. Consequently, that interface place is a passive input node for the *R*-labelled receive node contained in the consumer sphere.

Pattern 15 (Consumer-Initiated Bi-Directional)

Problem: The consumer needs to initiate an exchange with the service provider. However, the provider is immediately responding to the opposing Sourcing domain. Such bi-directional information exchange must not endanger the correct termination of a Sourcing configuration.

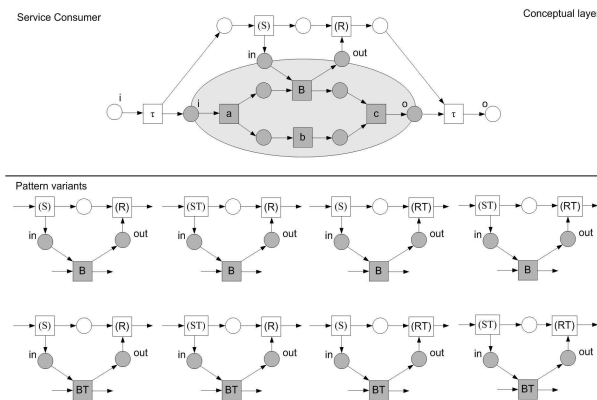


Fig. 16. Consumer-initiated bi-directional conjunction

Description: A sending conjunction node that is part of the in-house process outside of the consumer sphere, is connected to a port through which information is injected into Sourcing spheres. This port is connected to a bi-directional conjunction node that is

replicated in all spheres of to the same Sourcing configuration. The latter conjunction node responds with information that is sent through another port back to the domain of the service consumer. Finally, a receiving conjunction node located in the in-house process outside of the consumer sphere serves as a recipient for the information that is delivered through the second port.

Examples:

- A broker deals with a customer who wants to buy a house. While the process of finding a suitable house is on the way, in parallel a service provider deals with the financial matters related to providing a mortgage. For example, the service provider first needs to evaluate whether the customer of the broker is free of debts. At some point a house is found and the broker exchanges details about the price to the service provider. Based on the evaluated credit-worthiness, the provider needs to respond immediately whether a mortgage can be granted for purchasing the house. Based on the response, the broker goes ahead to offer the house or recommends that the customer needs to look for a less expensive house.
- The top of Figure 16 depicts an in-house process that contains a sending and receiving conjunction node. The first node carries a *S* label and the latter node a *R* label. The sending node has a passive output node that is an *in*-labelled interface place, which denotes an information exchange into a consumer sphere takes place. This interface place serves as a passive input node for the bi-directional conjunction node in the depicted consumer sphere. A contained *B*-labelled node produces an immediate response by placing a token in an *out*-labelled interface place. The latter interface place is a passive input node for the *R*-labelled receive node contained in the in-house process. Since the *R* and *S*-labelled conjunction nodes are positioned outside of the sphere, the provider's refined sphere does not contain equally labelled conjunction nodes. Instead the *in* and *out*-labelled interface places are part of all spheres involved in the Sourcing configuration. Consequently, a *B* or *BT*-labelled conjunction node is part of the refined sphere.

Finally, it needs to be stated that all four conjunction patterns are exposed to the same forces. When a conjunction construct is chosen, the performance characteristics of an exchange during enactment is influenced by incorporated monitorability patterns. Depending on the conjunction direction, the *out* and *in*-labelled interface places need to be linked with monitorability arcs. If messaging is applied, token propagation (see Pattern 4) is suitable and if polling is applied, token takeover (see Pattern 8) is appropriate. However, such a monitorability choice influences the speed of conjunction enactment. If token takeover is used, the defined polling period potentially postpones the conjunction until the next polling interval starts. If token propagation is chosen, the conjunction across Sourcing domains is started immediately. However, it is possible that the *R*, *RT*, or *B*-labelled conjunction node is not yet enabled for responding to the conjunction exchange.

Let us assume equally labelled conjunction nodes are part of the consumer sphere, the respective contractual spheres, and the refined sphere. If these active nodes are linked for life-cycle monitorability (see Pattern 7 and Pattern 11) then conjunction construction fails during build time if the lower level life-cycles can't be semantically matched. Furthermore, just as in the case of linking *out* and *in*-labelled interface places,

the speed of enacting conjunction nodes varies depending on the type of employed monitorability links.

After presenting pattern specifications for the Sourcing dimensions contractual visibility, monitorability, and conjunction, the following section presents a case study for Sourcing where the patterns are applied. The case study is input for the development of a proof-of-concept prototype in the CrossWork [1] project that focusses on the automobile industry.

9 Applying the Sourcing Patterns in a Case Study

In the automobile industry, OEM's have several tiers of suppliers that agree to deliver parts collaboratively. Such a constellation resembles a pyramid where the OEM at the top spends considerable time and effort on aligning first and second tier suppliers for achieving the desired service provision. Thus, in the automobile industry there exists a need for a more efficient establishment of service provision across multiple tiers of supply chain.

In this case study, a truck-producing company is a service consumer that does not build all parts of the trucks by itself. Instead it has several suppliers of components that are united in a regional automobile cluster of service providers. Within that cluster the service provider *A* functions as a unique communication party for the entire cluster to the truck producer. Once an order for service provision is given to party *A*, a decomposition of the service takes place within the automobile cluster. Thus, with the agreement of the truck producer, several members of the cluster commit themselves to provide parts of the overall service. Henceforth, the suppliers of the automobile cluster are termed service providers.

The water tank itself consists of a water-tank body, a grommet, a motor pump, a dispenser, and a sealing ring. Provider *A* receives the order for the entire automobile cluster and organizes the distribution of the production of water-tank parts to partners of the automobile cluster. Thus, it is decided the water-tank body and the grommet are produced by service provider *B*, the motor pump and the sealing ring are produced by service provider *C*, and the dispenser is produced by service provider *D*. Next, this water-tank production process is translated into a Sourcing configuration in the following subsection. Furthermore, provider *A* takes of the tasks of initially preparing the order and assembling the finished parts into the water tank before it is delivered to the OEM.

After presenting the context of the case study, the remaining subsections are organized as follows. First, the case study is translated into an overall Sourcing configuration where the collaborating parties of the automobile cluster are organized into a virtual-enterprize network for the OEM. Next, for sake of brevity only a subset of spheres is chosen for a detailed depiction that shows how the patterns of this paper are applied to improve the design time and the analysis of a Sourcing configuration. This pattern application is split in two parts. First, it is demonstrated how patterns are instrumental during the design phase of Sourcing spheres followed by an analysis phase for which the Sourcing patterns of this paper are equally be instrumental. Finally, the technology

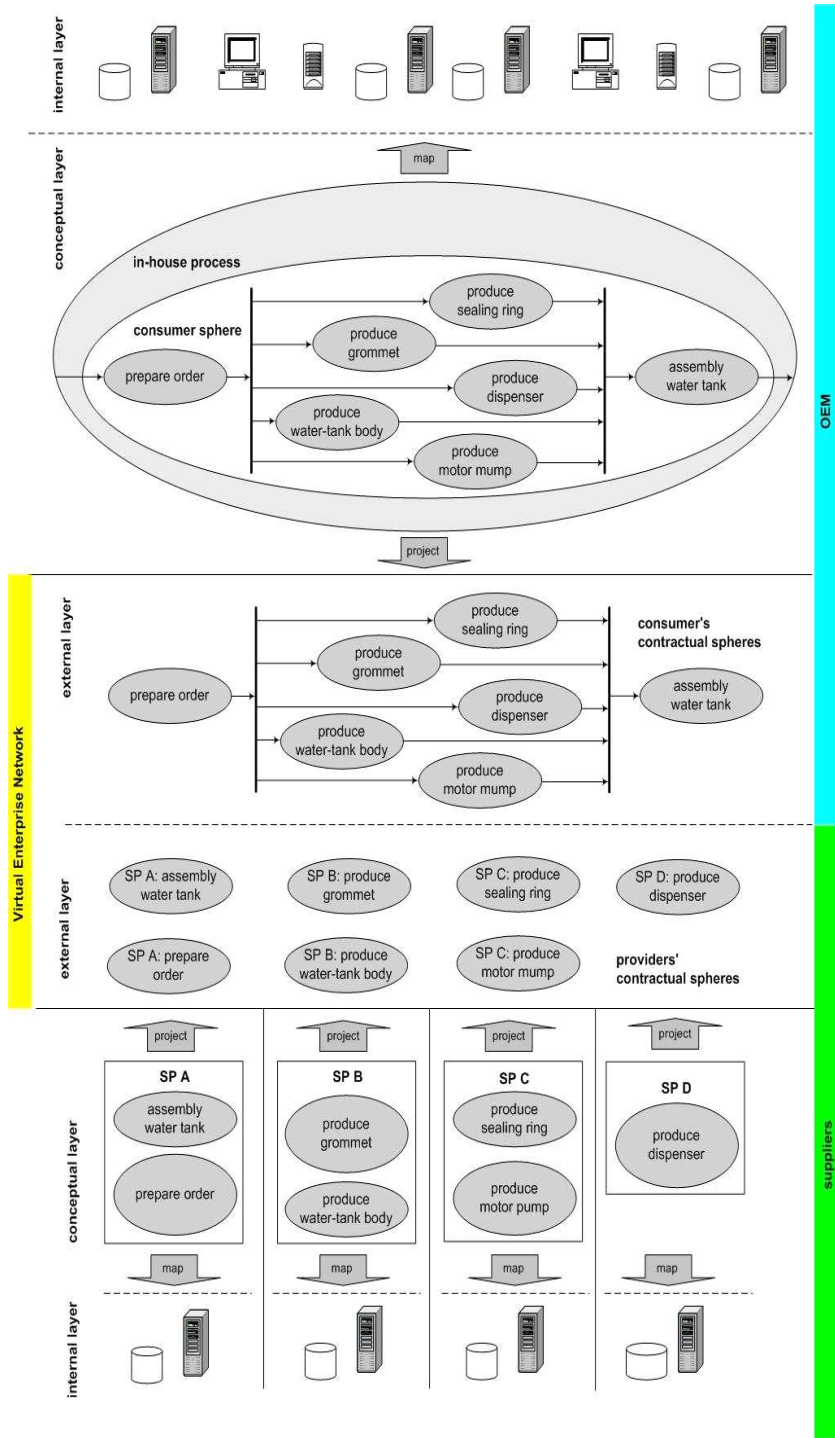


Fig. 17. The Overall Sourcing configuration for Sourcing a water-tank production process

employed in the proof-of-concept prototype for this water-tank case study is presented in the last subsection.

9.1 The Overall Sourcing Configuration

On the external level of Figure 17, several contractual spheres are depicted, which the consumer intends to source. From the consumer's perspective, first the order must be prepared before all respective parts of the water tank are produced in parallel branches. After the parallel branches are completed, the assembly takes place. The contractual spheres of the external level's virtual-enterprize network are translated to nested consumer spheres in one encapsulating consumer sphere. This consumer sphere is embedded as a subnet of the in-house process on the truck-producer's conceptual level. Thus, the in-house process is designed for building the entire truck of which the consumer sphere is a demarcation that focuses on building the water tank.

The external level of Figure 17 also shows the providers' contractual spheres. In the grey ellipses, labels indicate which collaborating party takes over what service provision. In order to achieve consensus on the external level, it is required that the contractual spheres of the virtual-enterprize network and the providers' contractual spheres match in content, i.e. contain equal tasks embedded in the same control-flow structure. Furthermore, a consensus must exist about conjoinment and the extent of monitorability. The providers have refined spheres on the respective conceptual levels, which is indicated in Figure 17 by bigger sized ellipses than their respective contractual spheres on the external level. Finally, for the OEM and the service providers of the automobile cluster, Figure 17 depicts internal levels that contain icons representing legacy systems. In a Sourcing configuration such legacy systems may be extended with a service-oriented architecture so that parts of it are represented in wrapping services that are addressable by the conceptual-level processes.

For sake of brevity it is not possible in this paper to offer a detailed presentation of all Sourcing spheres that Figure 17 depicts as grey shaded ellipses. Still, one subset of Sourcing spheres is chosen for a detailed study of pattern application. Concretely, the spheres of the conceptual and external levels that are related to the production of a motor pump are chosen for a deeper exploration of pattern application. To remain consistent with the modelling notation of the pattern specifications of this paper and point out the used patterns, the detailed spheres are depicted in a Petri-net format. Note, the detailed depiction of Figure 17 shows the end state of the Sourcing configuration. The figure does not reveal information about the sequence of interaction between the OEM and the providers that results in the depicted Sourcing spheres. Such patterns of interaction are not the focus of this paper.

After providing an overall description of the water-tank Sourcing configuration, the following subsections describe how patterns are used for the design and analysis phases of a Sourcing configuration.

9.2 Using Patterns in Sourcing-Configuration Design

As stated before, one subset of related spheres depicted in Figure 17 is taken for discussing pattern application. The conceptual level of the service consumer in Figure 18

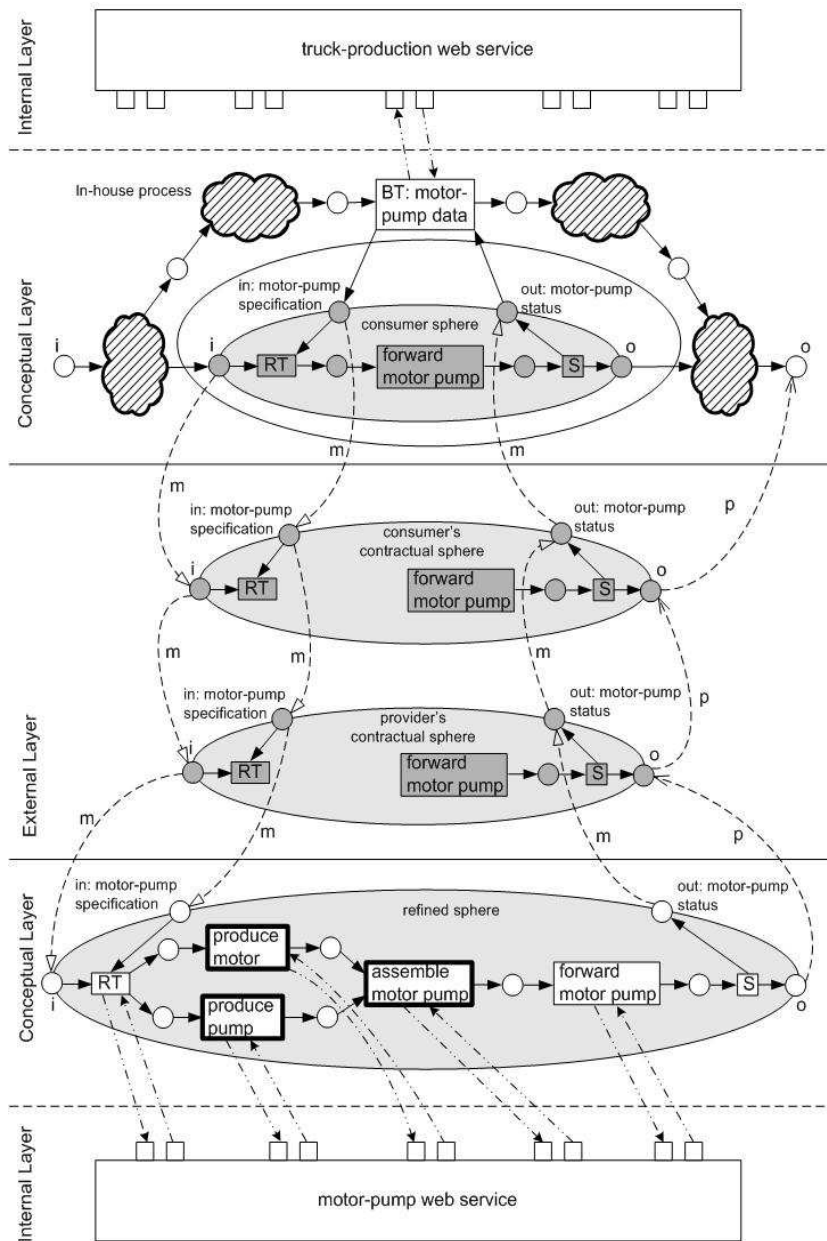


Fig. 18. Related Sourcing spheres in detail

shows a consumer sphere that is a subnet of an in-house process. Depicted clouds mean the figure abstracts from details of the in-house process. There is only one node depicted that interacts with the ports of the consumer sphere. This interacting node is a bi-directional conjunction node (see Pattern 14) that delivers a motor-pump specification to the *in*-labelled interface place of the consumer sphere after withdrawing such information from the web service of the internal level.

In the consumer sphere a receive task accepts the motor-pump specification. A receive task (see Figure 12) contains a lower level life cycle and requires resource involvement, e.g., for accepting and completing. Next, a task is contained for forwarding the finished motor pump. That way the truck producer determines which forwarding company the service provider needs to use if it is assumed the consumer sphere exists before the refined sphere and the contractual spheres. Finally, a send transition returns the status of the motor-pump to the bi-directional conjunction node via the *out*-labelled interface place. Note, a send transition (see Figure 12) does not involve resource involvement and it doesn't have a lower-level life cycle. Instead the node fires immediately when enabled. The service consumer chooses to use a bi-directional conjunction pattern because the in-house process is not supposed to carry on when the returned quality data isn't satisfactory.

The consumer sphere only defines nodes for exchanging business information and for determining who needs to forward the produced motor pump. While the service consumer intends to project this process content to the external level, the consumer is aware that more process nodes are required for producing the motor pump. At the same time the consumer can indicate where such corresponding nodes may be inserted. Thus, a grey-box pattern (see Pattern 3) is applied where a subset of the consumer sphere is projected to the contractual sphere on the external level.

The service provider can fill the void in the consumer spheres in the refined sphere with the required additional motor-pump production steps. This scenario is depicted on the external level of Figure 18 where the consumer's contractual sphere shows only a subset of nodes contained in the consumer sphere. A void exists between the receive task for the motor-pump specification and the task that specifies which forwarding company the service provider needs to use. Furthermore, the external level of Figure 18 also shows the provider's contractual sphere that contains the same set of equally labelled nodes embedded in the same control-flow constructs. Thus, a consensus is depicted that may be preceded by a possibly complex negotiation procedure between the collaborating parties. An investigation of such negotiation procedures is not focus of this paper.

On the conceptual level of the service provider, the contractual sphere is refined by additional nodes. Since a grey box contractual-visibility pattern is used, the provider has freedom to complete the content of the contractual-sphere process in the refined sphere. Differently to a white-box contractual pattern (see Pattern 1), the grey-box contractual pattern does not require adherence to projection inheritance (see Subsection 4.2). In the motor-pump spheres the OEM does not dictate in which way the production steps should be defined. Figure 18 shows boldly lined tasks that represent inserted nodes in the refined sphere. Thus, after receiving the motor-pump specification, the motor and the pump are produced in parallel branches before they are assembled in a joining task. Next, the ready motor pump is forwarded as defined by the truck producer and

subsequently quality data is transferred in a document about the motor-pump status to the domain of the service consumer. The tasks focusing on producing and forwarding the motor-pump interact with a web service of the provider.

For transitively linking the refined sphere with the consumer sphere via the external level, two monitorability patterns are used, namely token messaging (see Pattern 5) and token propagation (see Pattern 4). In the water-tank case study it is assumed that selecting such monitorability patterns is determined by the heterogenous system environment that are used in the domains of the collaborating parties. Token messaging is used for connecting the *i*-labelled interface places. For an enactment application of the Sourcing configuration token messaging means once the enactment of the in-house process has reached the consumer sphere, such a state is messaged across the organizational domains and the enactment of the provider's refined sphere commences. Token messaging is also used for connecting *in* and *out*-labelled interface places for exchanging the motor-pump specification and the status report across organizational domains. Finally, given the system infrastructure of the collaborating parties, token propagation is employed for connecting the *o*-labelled interface places of the refined sphere and the consumer sphere via the external level. In the Sourcing configuration of the water-tank case this means the enactment of the refined sphere is terminated and this event is communicated to the domain of the truck producer where the enactment of the next consumer sphere is starting.

9.3 Using Patterns in Sourcing-Configuration Analysis

The created Sourcing configuration of Figure 18 must be verified for correct termination before enactment. By doing so the likelihood of penalty payments because of failed service provision is reduced. Thus, the parties independently submit their respective processes to a trusted third party. The consumer submits the entire in-house process and the providers submit their refined spheres without having to disclose internal business details to the collaborating counterpart. The trusted third party "collapses" the Sourcing configuration by replacing all nested consumer spheres of the in-house process with the refined spheres. Next, the correct termination of the collapsed net is verified. For the Sourcing configuration of Figure 18 this verification fails. The reason is a deadlock contained in the processes that is caused by the bi-directional conjunction node. As this node needs to wait for the *S*-labelled conjunction node to fire, it can never be enabled. In fact, revisiting the specification of Pattern 14 makes it apparent that a provider-initiated bi-directional conjunction can not be applied for the nested consumer sphere of Figure 18 as the *RT*-labelled node is modelled before the *S*-labelled node.

After the evaluation of correct termination, the trusted third party sends the deadlock information back to the service consumer who has to remodel the in-house process. The changed in-house process depicted in Figure 19, has the bi-directional conjunction node replaced with a send task and a receive task. Since such conjunction nodes require resource involvement, it is ensured the in-house process can be halted if the quality feedback is not satisfactory. Consequently, a repeated termination verification by the trusted third party is successful, provided no deadlocks are caused by other parts of the Sourcing configuration.

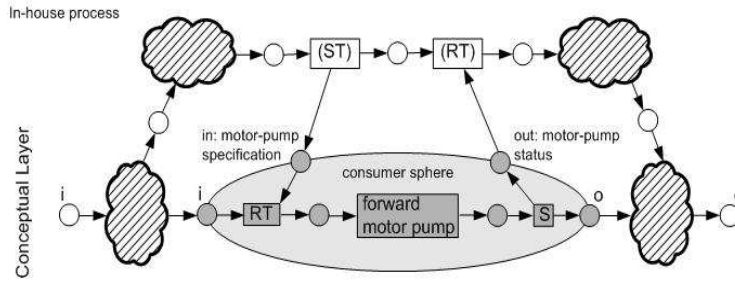


Fig. 19. Corrected in-house process

After showing how Sourcing patterns are applied in one set of spheres that belong to the water-tank case study for the CrossWork proof-of-concept prototype, the following subsection discusses in further detail the employed technology. That way the feasibility of tooling in a Sourcing context is demonstrated.

9.4 Employed Technology

By using service-oriented computing technology, the conceptual-level processes are harmonized on the external level of a Sourcing configuration by middleware. Furthermore, it is mentioned in Subsection 9.1 that legacy systems of collaborating parties need to be extended with a service-oriented architecture in order to be integrable in a Sourcing configuration. Thus, Figure 18 shows on the internal levels web services in the consumer and provider domain. These web services represent a bridge from the nodes of the conceptual levels to the respective legacy systems of the collaborating parties. In the water-tank case every collaborating party has one web service only for wrapping legacy systems.

For the external level of the water-tank case study, one file is defined in eSML [36] (electronic Sourcing Markup Language). The Sourcing patterns that are specified in this paper are all contained in eSML as elements and attributes of the language schema. Furthermore, eSML enables contracting as it is defined on top of ECML [13] (Electronic Contracting Markup Language) and covers the control-flow perspective by adopting XRL [33]. CrossWork has adopted XRL and its enactment application called XRL/flower [34, 35, 47]. Data-flow is realized by including in the eSML schema earlier specified patterns [44] of this perspective. The resource perspective of eSML is represented by adopting a merged model [10, 32, 43] that permits the capture of organizational and production facts such as machine data, material, space, and so on. These facts are instrumental for achieving an intelligent distribution of tasks, i.e., by paying attention to resource availability and limitations.

For all conceptual levels of the water-tank case study BPEL [19] is used. Thus, the transitions with *RT* labels of the conceptual level in Figure 18 stand for *receive* nodes in BPEL and an *S*-labelled transition for a *reply*. A *BT*-labelled transition is in BPEL an *invoke* node with contained *inputVariable* and *outputVariable* definitions. Transitions

with other labels on the conceptual level are simple invoke nodes. In the spheres on the external-level the *RT*-labelled transitions represent in eSML *receive_transition* nodes and *S*-labelled transition are *send_transition* nodes. All other transitions are *task* nodes in eSML. For adhering to Petri-net notation, the detailed depictions in the Sourcing spheres of Figure 18 contain states. Thus, the BPEL and eSML processes of the external and conceptual levels in the water-tank case study are not precisely represented by the detailed depictions of Figure 18. Still, as far as it is currently possible, the patterns of earlier sections of this paper are applied in the water-tank case study.

By choosing BPEL for the conceptual levels, enactment technology is readily available. However, BPEL is primarily intended for web-service orchestration and doesn't offer a comparable expressive power in differing perspectives such as Sourcing, control-flow, data-flow, resources, and so on, that are relevant for the creation of Sourcing configuration. Consequently, the application of eSML on the external level is drastically limited to the intersection of both languages. Note, the latter language is specifically designed for inter-organizational business process collaboration and supports all Sourcing patterns of this paper. This limitation means that only a subset of the eSML schema is applicable for the water-tank case, i.e., the elements and attributes that can either directly or with some approximation be mapped to BPEL elements and attributes.

Sourcing incorporates support for verifying the correctness of control-flow termination. Collaborating parties don't want to see the business process details of the Sourcing configuration in Figure 17 revealed to collaborating counterparts. For analyzing control-flow properties, the "collapsed" net of the Sourcing configuration must be mapped to a format that Woflan [46] understands as a trusted third party. In CrossWork this trusted third party is represented by a software agent. By performing a priori control-flow evaluation, the enactment phase of the Sourcing configuration is more likely to succeed. It is planned to extend the verification capabilities of Woflan to other perspectives such as data-flow and resource.

The case study of this section that applies the concept of Sourcing and patterns specified in this paper, shows how inter-organizational business process collaboration can be realized. However, the case study raises the question which scope exists for extending and enhancing the concept of Sourcing. The following section attempts to deal with this question.

10 Extending the Sourcing Concept

Based on the concept of Sourcing, scope for follow-up research is given. Other relevant perspectives for further research are data flow, resource, and exception and compensation handling. While patterns for an intra-organizational data-flow and resource perspective have been proposed [43, 44], the concept of Sourcing is indispensable for explore inter-organizational patterns for the data-flow and resource perspective on an inter-organizational level. The same is true for managing exception and compensation handling for Sourcing configurations that span across organizational boundaries.

Besides the Sourcing dimensions, more perspectives for the domain of DIBPM need to be explored that create additional, multi-dimensional spaces for other perspectives. For example, an interaction perspective is focusing on the way how a service consumer

and provider create a Sourcing configuration. Besides out-sourcing, where a consumer projects a sphere to an external level and the provider agrees to provide a refined sphere, other forms of build-time interaction exist based on the flexible concept of Sourcing. For example, the reverse scenario of in-sourcing starts with a sphere of a service provider that is projected to the external level before the consumer accepts and integrates the sphere in its own in-house process on the conceptual level. Furthermore, the Sourcing counterparts may pursue a public-to-private approach [11] where the interaction between Sourcing counterparts starts with the contractual spheres of the external level followed by the creation of conceptual-level processes. A reversed interaction direction from private-to-public starts with conceptual-level processes of the Sourcing counterparts that are forged on an external level into a Sourcing configuration.

Moreover, the utilization of Sourcing spheres in a three-level framework forms a base for exploring exception handling and compensation that is part of a transaction concept that needs to reflect e-business requirements. Since Sourcing is intended for commercial inter-organizational collaboration, it is essential to lay a foundation for a concept that not only supports a negotiation-process during the formation of a Sourcing configuration but also ensures enactment safety. By using spheres for providing and consuming services, it is feasible to equip a Sourcing configuration with an inter-organizational business-transaction concept. A transaction model for inter-organizational business collaboration termed X-transaction [22] is adaptable for Sourcing. The use of spheres permits the introduction of necessary e-business specific atomicities [39], e.g., contractual atomicity, non-repudiation atomicity, payment atomicity, delivery atomicity. Furthermore, Sourcing can be combined with the ongoing project called XTC [27, 48] (eXecution of Transactional Contracted e-services), where the objective is pursued to develop a business-transaction framework that utilizes abstract transactional constructs to provide a generic foundation for the support of complex transactional services in contract-driven inter-organizational business interactions that rely on dynamically composed web services.

Finally, while the conceptual sections of this paper focus on bi-lateral contracting, the case study demonstrates that the concept of Sourcing is sufficiently flexible to cater for multi-lateral contracting. Therefore, future research shall investigate the scenario of multiple Sourcing spheres in one service consumer's conceptual-level process that are projected in one connected process to the external level. Thus, each Sourcing sphere of the service consumer's contractual process is related to a separate provider sphere contained on the external level. Accordingly, several conceptual levels for each respective service provider contain the corresponding refined spheres of a Sourcing configuration. However, in future research such bi-lateral contracting requires adjustments with respect to the control-flow properties (see Subsection 4) of the service consumer's processes on the conceptual and external level.

11 Conclusion

For an exploration of Sourcing, this paper pursues a top-down method of pattern discovery. Sourcing uses a three-level business process framework to manage the conceptual, business, and technological complexity involved in inter-organizational collabora-

tion for commercial purposes. Furthermore, the paper informally presents control-flow properties of a Sourcing configuration, which is a prerequisite for structurally matching processes of a service consumer and provider. These properties include provisions that allow a service provider to extend a business process with additional tasks that are opaque for the service consumer without violating the desired enactment behavior of the overall service provision.

The characteristics of Sourcing are explored in a top-down way based on a multi-dimensional, logical space with the dimensions contractual visibility, monitorability, and conjoinment. Further refining dimension values create a taxonomy for Sourcing-pattern specifications, which allows to deal with the inherent complexity of Sourcing in a technology independent way.

The patterns of this paper are used in the ongoing EU-project called CrossWork where Sourcing is an integral concept. In a work package of CrossWork an XML based language called eSML (electronic Sourcing Markup Language) has been developed. In ongoing case studies Sourcing scenarios from industrial partners of the CrossWork project are formulated in eSML. Currently a proof-of-concept prototype is under development as part of a deliverable for CrossWork that carries out these eSML-formulated Sourcing scenarios in a web-based way.

The case study shows that Sourcing is a suitable concept for matching service consuming and providing business processes while allowing each party to keep essential business-process parts hidden from each other. Tool support is available for verifying before enactment the correct termination of processes that are part of a Sourcing configuration. Furthermore, without disclosing process details of internally refined service provision, tool support is instrumental for assuring a consumer that a provider adheres to the agreed upon service without imposing fixed, standardized routing on the latter party.

References

1. CrossWork: Cross-Organisational Workflow Formation and Enactment. <http://www.crosswork.info/>.
2. IBM MQSeries workflow. <http://www-4.ibm.com/software/mqseries/workflow>.
3. W.M.P. van der Aalst. Inheritance of Interorganizational Workflows: How to Agree to Disagree Without Loosing Control? BETA Working Paper Series, WP 46, Eindhoven University of Technology, Eindhoven, 2000.
4. W.M.P. van der Aalst and T. Basten. Inheritance of workflows: An approach to tackling problems related to change. *Theoretical Computer Science*, 270(1-2):125–203, 2002.
5. W.M.P. van der Aalst, M. Dumas, A.H.M. ter Hofstede, and P. Wohed. Pattern-Based Analysis of BPML (and WSCI). *QUT Technical report*, (FIT-TR-2002-05):487–531, 2002.
6. W.M.P. van der Aalst and K.M. van Hee. *Workflow Management: Models, Methods, and Systems*. MIT press, Cambridge, MA, 2002.
7. W.M.P. van der Aalst, A.H.M. ter Hofstede, B. Kiepuszewski, and A.P. Barros. Workflow Patterns Home Page. <http://www.tm.tue.nl/it/research/patterns/>.
8. W.M.P. van der Aalst, A.H.M. ter Hofstede, B. Kiepuszewski, and A.P. Barros. Advanced Workflow Patterns. In O. Etzion and P. Scheuermann, editors, *7th International Conference on Cooperative Information Systems (CoopIS 2000)*, volume 1901 of *Lecture Notes in Computer Science*, pages 18–29. Springer-Verlag, Berlin, 2000.

9. W.M.P. van der Aalst, A.H.M. ter Hofstede, B. Kiepuszewski, and A.P. Barros. Workflow Patterns. *Distributed and Parallel Databases*, 14(3):5–51, 2003.
10. W.M.P. van der Aalst, A. Kumar, and H.M.W. Verbeek. Organizational modeling in uml and xml in the context of workflow systems. In H. Haddad and G. Papadopoulos, editors, *Proceedings of the 18th Annual ACM Symposium on Applied Computing (SAC 2003)*, page ??, 2003.
11. W.M.P. van der Aalst and M. Weske. The P2P approach to Interorganizational Workflows. In K.R. Dittrich, A. Geppert, and M.C. Norrie, editors, *Proceedings of the 13th International Conference on Advanced Information Systems Engineering (CAiSE'01)*, volume 2068 of *Lecture Notes in Computer Science*, pages 140–156. Springer-Verlag, Berlin, 2001.
12. G. Alonso, S.U. Fiedler, C. Hagen, A. Lazcano, H.Schuldt, and N. Weiler. Wise: Business to business e-commerce. In *Proceedings of the 9th International Workshop on Research Issues on Data Engineering*, pages 132–139, Sydney, Australia, 1999.
13. S. Angelov. *Foundations of B2B Electronic Contracting*. Dissertation, Technology University Eindhoven, Faculty of Technology Management, Information Systems Department, 2006.
14. S. Angelov and P. Grefen. The 4W framework for B2B e-contracting. *International Journal of Networking and Virtual Organizations*, 2(1):78–97, 2003.
15. K. Baina, K. Benali, and C. Godart. Dynamic interconnection of heterogeneous workflow processes through services. In R. Meersman, Z. Tari, and D.C. Schmidt, editors, *OTM Confederated International Conferences, CoopIS, DOA, and ODBASE 2003*, number 2888 in *Lecture Notes in Computer Science*, pages 444–461, Catania, Sicily, Italy, 2003. Springer Verlag, Berlin.
16. A. Barros, M. Dumas, and A.H.M. ter Hofstede. Service interaction patterns. In W.M. P. van der Aalst and F. Curbera B. Benatallah, F. Casati, editors, *Business Process Management: 3rd International Conference, BPM 2005*, number 3649 in *Lecture Notes in Computer Science*, pages 302–318, Nancy, France, 2005. Springer Verlag, Berlin.
17. BEA Systems, Intalio, SAP AG, Sun Microsystems. *Web Service Choreography Interface (WSCI) 1.0 Specification*. <http://www.sun.com/software/xml/developers/wsci/>, 2003.
18. BPML.org. *Business Process Modeling Language (BPML) version 1.0*. Accessed August 2003 from www.bpmi.org, 2003.
19. F. Curbera, Y. Golland, J. Klein, F. Leymann, D. Roller, S. Thatte, and S. Weerawarana. *Business Process Execution Language for Web-Services*. <http://www-106.ibm.com/developerworks/library/ws-bpel/>, 2003.
20. E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Professional Computing Series. Addison Wesley, Reading, MA, USA, 1995.
21. P. Grefen. Service-Oriented Support for Dynamic Interorganizational Business Process Management. to appear, 2005.
22. P. Grefen, K. Aberer, Y. Hoffner, and H. Ludwig. CrossFlow: Cross-organizational Workflow Management in Dynamic Virtual Enterprises. *International Journal of Computer Systems, Science, and Engineering*, 15(5):277–290, 2001.
23. P. Grefen, H. Ludwig, and S. Angelov. A Three-Level Framework for Process and Data Management of Complex E-Services. *International Journal of Cooperative Information Systems*, 12(4):487–531, 2003.
24. Y. Hoffner, H. Ludwig, C. Gülcü, and P. Grefen. Architecture for Cross-Organizational Business Processes. In *Procs. 2nd Int. Workshop on Advanced Issues of E-Commerce and Web-Based Information Systems*, pages 2–11, Milpitas, CA, USA, 2005.
25. IBM. *Web Service Flow Language (WSFL) 1.0 Specification*. <http://www-3.ibm.com/software/solutions/webservices/pdf/WSFL.pdf>, 2003.

26. B. Kiepuszewski. *Expressiveness and Suitability of Languages for Control Flow Modelling in Workflows*. PhD thesis, Queensland University of Technology, Queensland University of Technology, Brisbane, Australia, 2002.
27. B. Kratz, T. Wang, J. Vonk, and P. Grefen. Flexible Business Transaction Composition in Service-Oriented Environments. BETA Working Paper Series, WP 154, Eindhoven University of Technology, Eindhoven, 2005.
28. J. Krogstie and H.D. Jorgensen. Interactive models for supporting networked organizations. In A. Persson and J. Stirna, editors, *Advanced Information Systems Engineering, 16th International Conference, CAiSE 2004*, number 3084 in Lecture Notes in Computer Science, pages 550–563, Riga, Latvia, 2004. Springer Verlag, Berlin.
29. A. Kumar, J. Wainer, and Z. Zhang. Meta-workflows and esp: A framework for coordination, exception handling and adaptability in workflow systems. In C. Bussler, D. Fensel, M.E. Orłowska, and J. Yang, editors, *Web Services, E-Business, and the Semantic Web, Second International Workshop, WES 2003*, number 3095 in Lecture Notes in Computer Science, pages 13–27, Klagenfurt, Austria, 2004. Springer Verlag, Berlin.
30. A. Lazcano, H. Schuldt, G. Alonso, and H. Schek. WISE: Process Based E-Commerce. *IEEE Data Engineering Bulletin*, 24(1), 2001.
31. F. Leymann and D. Roller. *Production Workflow: Concepts and Techniques*. Prentice-Hall PTR, Upper Saddle River, New Jersey, USA, 1999.
32. M. zur Muehlen. *Workflow-based Process Control: Foundation, Design, and Application of Workflow-driven Process Information Systems*. Advances in Information Systems and Management Science, 2002.
33. A. Norta. XRL Home Page. <http://is.tm.tue.nl/research/xrl/>.
34. A. Norta. XRL/flower Home Page. <http://is.tm.tue.nl/research/xrl/flower>.
35. A. Norta. Web supported enactment of petri-net based workflows with xrl/flower. In J. Cortadella and W. Reisig, editors, *Proceedings of the 25th International Conference on the Application and Theory of Petri Nets 2004*, number 3099 in Lecture Notes in Computer Science, pages 494–503, Bologna, Italy, 2004. Springer Verlag, Berlin.
36. A. Norta. eSML (electronic Sourcing Markup Language) in: R. Eshuis and I. Stalker, Cross-Work: Requirements and concepts for agent-based workflowconfiguration, Work Package 2: Workflow Modelling, Task 2.2: Agent-Based Workflow Configuration, 2005.
37. A. Norta and P. Grefen. Sourcing: Introducing a Concept for Supporting Business-to-Business Collaboration. to appear, 2005.
38. Queensland University of Technology. YAWL Home Page. <http://www.yawl-system.com>.
39. M.P. Papazoglou. Web Services and Business Transactions. *World Wide Web: Internet and Web Information Systems*, 6:49–91, 2003.
40. W. Reisig and G. Rozenberg, editors. *Lectures on Petri Nets I: Basic Models*, volume 1491 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, 1998.
41. W. Reisig and G. Rozenberg, editors. *Lectures on Petri Nets II: Applications*, volume 1492 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, 1998.
42. IBM Research. Crossflow architecture description, Technical report, ESPRIT Crossflow EP 28653, 1999.
43. N. Russell, A.H.M. ter Hofstede, D. Edmond, and W.M.P. van der Aalst. Workflow Resource Patterns. *QUT Technical report*, forthcoming.
44. N. Russell, A.H.M. ter Hofstede, D. Edmond, and W.M.P. van der Aalst. Workflow Data Patterns. *QUT Technical report*, (FIT-TR-2004-01), 2004.
45. S. Thatte. *XLANG: Web Service for Business Process Design*, 2003.
46. H.M.W. Verbeek, T. Basten, and W.M.P. van der Aalst. Diagnosing Workflow Processes Using Woflan. *The Computer Journal, British Computer Society*, 44(4):246–279, 2001.

47. H.M.W Verbeek, A. Hirnschall, and W.M.P. van der Aalst. XRL/Flower: Supporting inter-organizational workflows using XML/Petri-net technology. In C. Bussler, R. Hull, S. McIlraith, M.E. Orłowska, B. Pernici, and J. Yang, editors, *Web Services, E-Business, and the Semantic Web*, CAiSE 2002 International Workshop, WES 2002, Toronto, Canada, pages 93–109. LNCS Springer, May 2002.
48. T. Wang. Towards a transaction framework for contract-driven service-oriented business processes. In A. Hanemann, editor, *Proceedings of the IBM PhD Student Symposium at 3rd International Conference on Service Oriented Computing (ICSOC2005)*, pages 43–48, 2005.
49. P. Wohed, W.M.P. van der Aalst, M. Dumas, and A.H.M. ter Hofstede. Analysis of web services composition languages: The case of bpel4ws. In I.Y. Song, S.W. Liddle, T.W. Ling, and P. Scheuermann, editors, *22nd International Conference on Conceptual Modeling (ER 2003)*, number 2813 in Lecture Notes in Computer Science, pages 200–215, Chicago, Illinois, 2003. Springer Verlag, Berlin.
50. J. Zdravkovic and P. Johanesson. Cooperation of processes through message level agreement. In A. Persson and J. Stirna, editors, *Advanced Information Systems Engineering, 16th International Conference, CAiSE 2004*, number 3084 in Lecture Notes in Computer Science, pages 564–579, Riga, Latvia, 2004. Springer Verlag, Berlin.