Facilitating Cross-Organisational Workflows with a Workflow View Approach

Karsten A. Schulz

SAP Australia Pty Ltd.
Corporate Research
Brisbane, QLD 4000 Australia
ka.schulz@sap.com

Maria E. Orlowska

School of Information Technology and Electrical Engineering
The University of Queensland
QLD 4072 Australia
maria@csee.uq.edu.au

Abstract

Interconnecting business processes across systems and organisations is considered to provide significant benefits, such as greater process transparency, higher degrees of integration, facilitation of communication, and consequently higher throughput in a given time interval. However, to achieve these benefits requires tackling constraints. In the context of this paper these are privacy-requirements of the involved workflows and their mutual dependencies. Workflow views are a promising conceptional approach to address the issue of privacy; however this approach requires addressing the issue of interdependencies between workflow view and adjacent private workflow. In this paper we focus on three aspects concerning the support for execution of cross-organisational workflows that have been modelled with a workflow view approach: (i) communication between the entities of a view-based workflow model, (ii) their impact on an extended workflow engine, and (iii) the design of a cross-organisational workflow architecture (CWA). We consider communication aspects in terms of state-dependencies and control flow dependencies. We propose to tightly couple private workflow and workflow view with state dependencies, whilst to loosely couple workflow views with control flow dependencies. We introduce a Petri-Net-based state transition approach that binds states of private workflow tasks to their adjacent workflow view-task. On the basis of these communication aspects we develop a CWA for view-based cross-organisational workflow execution. Its concepts are valid for mediated and unmediated interactions and express no choice of a particular technology. The concepts are demonstrated by a scenario, run by two extended workflow management systems.

Key words: Cross-Organisational Workflow, Workflow View, Workflow Engine, Cross-Organisational Workflow Architecture.

1 Introduction and Problem Description

1.1 Motivation

We observe that the notion of Business Process Management (BPM) is receiving increasing attention from analysts, customers, and solution providers. BPM is considered to provide comprehensive answers to the problem of application component, application, system, and business integration by providing higher-level constructs that are able to reflect contexts, involved entities, and their dependencies. Particular in the context of cross-organisational interactions (a.k.a. Business-to-Business), we observe that a particular context requires the involved partners to adapt for the purpose of the interaction. This adaptation can, however, not necessarily be reflected in the partners' private (internal) business processes without inflicting their ability to interact with other partners in a different context. Imagine an automotive supplier that is providing parts to two different car manufacturers that prescribe a particular sequence of interaction. This means that process-oriented abstractions need to be modelled and tightly bound to their corresponding private business process. Furthermore, these have to be executed in equilibrium such that internal actions that affect the interaction with other partners are communicated, and that the coordination between the partners effects their private workflows, i.e., that they synchronise with each other.

Workflow views are considered a promising conceptual approach to selectively hide details of private workflows, whilst providing a process-oriented interface to facilitate the state-oriented communication between trading partners. However, with the introduction of workflow views we need to address the issue on how a workflow engine could support them and how this would affect the design of an underlying architecture that would facilitate the cross-workflow-system communication. Workflow interoperability standards, such as BPEL4WS [17], and WSCI [13] provide important constructs to model interactions of web services. However, they neither address a model-oriented relationship between private workflows and their publicly visible abstractions nor do they model the interaction between abstract processes. It is further not the focus of these specification to propose how such a model could be supported by a runtime engine. In this paper we therefore consider a workflow management system (WfMS) and conceptualise a supporting architecture that enables execution of private workflows in a cross-organisational context without compromising the privacy and contingency of private workflows, which reflect best business practice, and provide a scalable visibility to structural information of private workflows to trading partners.

This paper is organised as follows: In Section 1 be motivate our work, clarify our understanding on cross-organisational workflow, elaborate on related work, and introduce a distributed workflow model. In Section 2 we consider architectural aspects, a scenario, and unmediated and mediated interactions. In Section 3 we particularly focus on an approach to tightly bind a private workflow to its corresponding abstract workflow and investigate how this approach behaves during execution time. In Section 4 we discuss a supporting cross-organisational workflow architecture and introduce its constituent entities. Section 5 presents an extended workflow management system for cross-organisational workflows, called Nehemiah. We conclude and mention future work in Section 6.

1.2 On Cross-Organisational Workflow Types

In order to position the particular architectural issues in cross-organisational workflow management we need to investigate and clarify our understanding of this term. We observe the involvement of an external party which is interacting within the context of a task of a workflow or within the context of a workflow. We therefore consider cross-organisational / cross-system workflows, which we classify into two main types: **Distributed Workflows**, also known as composite workflows, and **Outsourced Workflows**.

1.2.1 Distributed Workflows

All activities or sub-workflows of the workflow of a party are implemented inside the scope of the workflow. The existing workflow is augmented by means of one or more activities or sub-workflows of an external workflow. In Figure 1 we consider a distributed workflow (coalition workflow) with the participation of the Partners *A* and *B*. Partner A owns and implements tasks t_2, t_4, t_5, which are underlaid in dark-grey in the figure. Partner B owns and implements tasks t_1, t_3, t_6, which are underlaid in light grey in the figure.

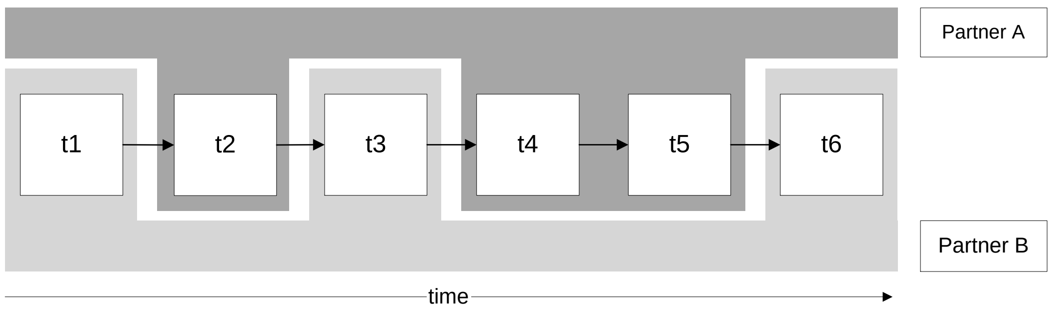


Fig. 1. Distributed Workflow

In Figure 2 we analyse this distributed workflow in greater detail. Because both workflows from Partner A and B are pre-existing, they are complete in the sense that their respective tasks are linked by dependencies; these are $d(t_2,t_4)$, $d(t_4,t_5)$, $d(t_1,t_3)$ and $d(t_3,t_6)$.

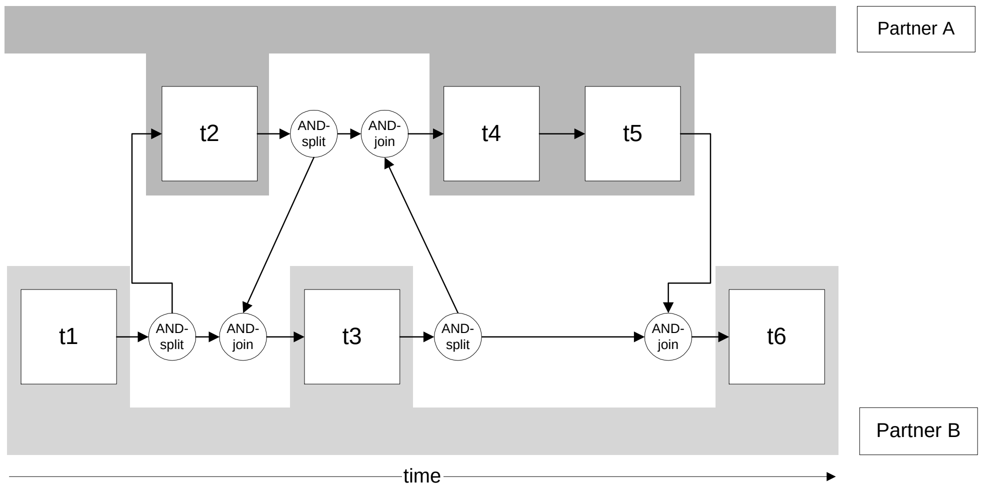


Fig. 2. Distributed Workflow with Synchronisation Tasks

As the workflows from Partners A and B are run by different systems, it is important for the systems to coordinate their interactions, e.g., by means of routing tasks and dependencies. In the example, t_4 needs to wait until t_2 AND t_3 are completed. Therefore, Figure 2 also depicts the routing tasks that are required to ensure order preservation and co-ordination of tasks $t_1 \dots t_6$. It is necessary to apply the routing tasks, which are *AND-splits* and *AND-joins* in order to synchronise the flow of work within a partner's organisation with the flow of work outside. In the figure, each outgoing dependency requires the emitting task to add an AND-split, whilst each incoming dependency requires the receiving task to add an AND-join.

1.2.2 Outsourced Workflows

In Outsourced Workflows, one or more activities or sub-workflows of an existing workflow are implemented outside of the scope of the workflow by an external task or workflow. The existing activities or sub-workflows in the workflow are placeholders for external activities or sub-workflows. Figure 3 shows the general principles of an outsourced workflow model.

In the figure, partner B's tasks t_{12} and t_{14} are proxies for partner A's tasks t_{13}, t_{15}, and t_{16} that implement them. Tasks t_{12} and t_{14} communicate with partner A's implementing tasks during their lifecycle. This is transparent for workflow management system that performs the workflow of partner B because the outsourcing acts exactly the same as a performing agent.

With regards to our architectural considerations we propose to realise distributed workflows through control-flow dependencies, whilst outsourced workflows better

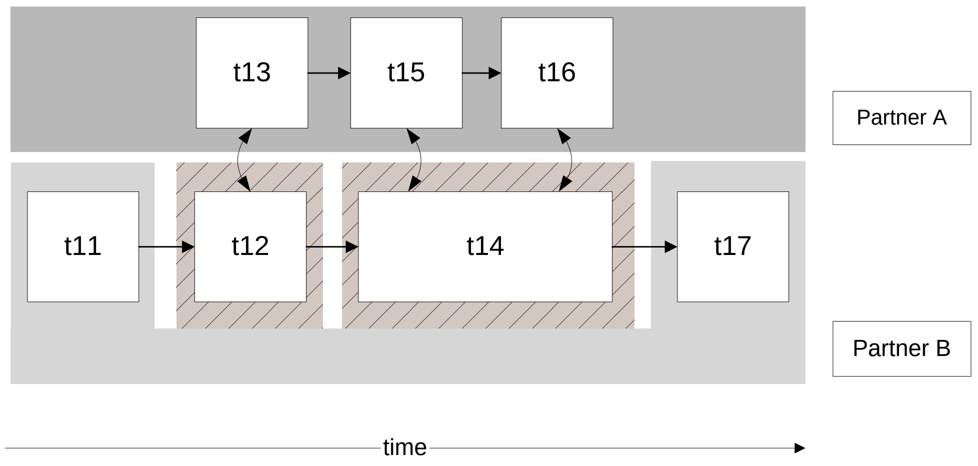


Fig. 3. Outsourced Workflow

communicate through state dependencies. We discuss this in the following sections and relate our proposed workflow model to the concepts of workflow distribution and outsourcing.

1.3 Related Work

The topic of architecting cross-organisational process interactions has received significant attention in the research community. Common to these approaches is that cross-organisational process modelling and architecting are inherently interweaved.

We observed that there are two contradicting key requirements for cross-organisational workflows. One is that interactions between business partners have to remain flexible to allow for change. The other is that this interaction is required to be robust and secure. The notion of separating partner-internal and partner-external process representation initially resulted in private-public models, such as described in [42], [26], and [29]. This concept has resulted in a number of standardisation activities, such as BPEL4WS [17], which supports the modelling of executable (private) and abstract (public) processes, and WSCI [13], which addresses the issue of interactions with web service in a single context (public).

Research Projects

The Vega Project (Virtual Enterprises using Groupware tools and shared Architectures) [42], [45], [46], developed a platform to support the interaction of users and IT systems from different organisations that form a virtual enterprise. The Vega platform consists of a data integration layer and a distributed workflow service. The workflow service is 2-tiered: A public workflow service manages points of partner interaction and sends messages to the local systems of the partner organisations. The messages are filtered by a special workflow gateway, which translates them into the particular organisation-internal format and protects local workflows. The Vega distributed workflow service follows a black-box approach in which the individual workflow models and the corresponding workflow implementations are not required to be made public to other partners in the virtual enterprise. It thus fully decouples systems that are involved in a cross-organisational workflow. A tool for the modelling of inter-organisational workflows is part of the service. The distributed workflow service also provides the capability to monitor cross-organisational workflows by allowing the partners to supply the level of monitoring data that they want to make available to the virtual enterprise. The implementation builds upon OMG's Workflow Management Facility Specification concept [18], [19] for inter-workflow management interfaces, incorporates existing workflow management systems, including SAP R/3, and also non process-oriented systems. Available services are selected at runtime according to their capabilities and their availability.

The Crossflow project [35] formulated an approach to architect cross-organisational business process interactions. The Crossflow architecture builds upon proxy gateways that protect or encapsulate the provider/consumer core services of the organisations. The architecture is workflow management system independent. Crossflow introduces business contracts to describe the business operation between partners. Crossflow requires direct communication between partner organisations to execute a joint business process. In Crossflow, there is no central instance that has a comprehensive view of the multi-organisational business processes. Elements of this research are described in [1]. Most notably this paper describes a high-level view on a service, distinguishing the four interfaces: *input* and *output* data flow and *control* and *notification* control flow. The authors assign a Quality-of-Service (QoS) attribute to services and base their algorithm for service selection on the abstract QoS parameter *cost*. This work is further described in [2] where the authors explore the outsourcing of service provision to business partners and the limitations on the side of the service requester to monitor the internal processing of the service provider. The authors then present a service model on the basis of continuous-time Markov chains, which is a stochastic process through different states in certain time epochs. The general idea is, that the service requester can obtain an understanding of the service provider by analyzing the service-provider's log data against the continuous-time Markov chain model.

Similar work has been carried out by van den Heuvel and Weigand [49]. The authors propose a contract model to represent formal and informal communication structures that are utilised to coordinate business workflows. The approach assumes workflows that are private within organisations. A contract exposes selected activities and links them to activities of other participants. The contract then specifies the messages that can be exchanged between them.

Riempp [36] builds upon shared repositories to interchange data between linked business processes. Riempp introduces an additional interface between Process Definition Tools of the involved business partners and assumes joint modelling of business process definitions. Riempp assumes the following three key characteristics of cross-organisational workflows:

(1) Autonomy of workflow management systems.
(2) Coordination through mutual agreements as opposed to coordination through a central instance.
(3) Heterogeneity of workflow management systems.

Riempp introduces the Widea Area Group flow System, WAGS. In WAGS, there is a distinction between public and private workflows. The publication of workflow entities that require external interaction is restricted to tasks of type *IN*. This restricts the interactions between workflow services to the chained and nested models of interoperability [6] and does not allow for fine-grained interaction between the trading partners. Riempp introduces modelling entities of type *CLUSTER* which support a continuum of joint modelling mechanisms, ranging from *publication and subscription* to *centralised modelling*.

Umar et al [48] introduce a general methodology for the analysis of virtual enterprises and their services, based on an electronic commerce approach. They propose a high-level platform for virtual enterprises and describe a set of required services, without providing a deeper analysis of them. The paper concludes with a proposition of research issues that are worth pursuing: (1) Dynamic service creation and -assembly, (2) support of mobility of users and partners, (3) customer care and support, (4) management of virtual enterprises, and (5) building of virtual enterprise services.

Chen et al [3] addresses the problem of interoperability aspects in cross-organisational workflow management. The paper describes a collaborative process manager for cross-organisational peer-to-peer workflows. A cross-organisational workflow is instantiated at each participating organisation. With n participants, this results in n identical workflow instances. These instances need to communicate to synchronise their operations. The challenge is that the change in the status of a task at the site of one participant needs to be communicated to all parties. There is no model-based hiding of private workflows.

The WISE project described by Alonso et al in [14] and by Lazcano et al in [12] is an integration effort where several known technologies were brought together in order to provide a coherent technological solution for inter-organisational workflows. Most notably, WISE introduces a centrally coordinated approach. A central workflow engine executes cross-organisational workflows. Subsystem adapters encapsulate the knowledge necessary for the interaction with external systems (protocols, parameters, interfaces, ...). In the WISE model, partner A does not communicate directly with partner B since this means that partner A needs implicit information about partner B's internal process, its state, actors, configuration and so forth. Therefore, the system dynamically establishes a communication channel between actors, i.e. context-specific collaboration is provided. WISE clearly distinguishes trading communities, virtual business processes and virtual enterprises and relates them as 1:n:1. A trading community represents all the participating companies in a virtual enterprise. Inter-organisational workflows interconnect individual partner workflows. Only the interfaces of the individual workflows are required to be known by the virtual enterprise (black-box approach). The WISE model corresponds to the approach taken by the VEGA distributed workflow service and extends it by an explicit service architecture to model virtual enterprise workflows.

Muth et al start in [16] with the problem of history management (monitoring) in a cross-organisational environment, such as a virtual enterprise. They argue to model history management and worklist management on top of a light-weight workflow management system. This WfMS collects information from other WfMS in the virtual enterprise and provides a single interface. This approach is similar to the Vega distributed workflow service.

Tagg et al [34] propose an interconnection model for virtual enterprises. Their approach is to extend the ISO/OSI reference model by adding layers 7a, 7b, 8, 9, 10, 11. These layers reflect the types of interaction between Virtual Enterprise (VE) participants from informal discussions (layer 11) to requests between computer systems across data communication facilities (layer 7b). The paper suggests that several existing technologies, such as WPDL, RosettaNet, SOAP, XML, etc. can be mapped to the extended ISO/OSI model. The authors distinguish formal and informal interactions between VE participants and argue that the interactions are determined by the lifecycle phase of a VE and the dominance of individual partners.

Johannesson and Perjons [33] present design principles for application integration. The authors state that application integration is fuelled by the move to process orientation in many organisations. The authors distinguish three major architectures for application integration: the point-to-point strategy where every application is directly connected to every other application is only applicable to a small number of application systems. There are $\frac{n(n-1)}{2}$ bi-directional connections between n application systems. As the number of application systems increases, the number of connections increases quadratic. A Message Broker architecture reduces this complexity down to n bi-directional connections by introducing a central message passing instance. Its main responsibilities are message routing and format conversion. To handle integration of applications efficiently, the authors propose a Process Broker. In addition to handling format conversions, the Process Broker also encapsulates the process logic for connecting applications. The authors argue that when all process logic resides in one place, it becomes possible to study, analyse, and change the processes coherently.

Johannsson's and Perjon's work neglects the topic of fine-grained workflow interactions but proposes an interesting approach for a methodology of application integration. It should be seen in conjunction with the work of Becker and zur Muehlen in [51]: The authors describe a classification framework for application granularity in workflow management systems. The authors distinguish three levels of granularity: coarse, medium and fine. The coarse application granularity means that a WfMS uses an agent (application) as whole, while only few application data is exchanged between the WfMS and the application. Mainly, control data, such as start/stop is exchanged - an application is invoked by the WfMS and returns a result and completes. The complexity of the business process is low. This is a model for the integration of legacy systems in a business process. The fine application granularity can be observed when the WfMS controls elementary functions of the application system. Here, the WfMS required detailed knowledge about the application system, thus the workflow is very complex. This way the WfMS takes over the internal control system's work of the application system. The medium application granularity can be observed when the WfMS invokes parts of application systems, such as transactions in an ERP-software or modules of a legacy system. Application data is exchanged between the WfMS and the application system. This puts higher demands to data consistency. The medium application granularity is most likely to be encountered in cross-organisational environments.

Standards

Workflow Standards form an important basis for cross-organisational research as they define common data models and communication protocols. The Workflow Management Coalition (WfMC) developed a common meta-model for describing workflow management systems, process definitions, and workflow management system interfaces. The industry standardisation activities of the WfMC have been documented in several standards documents, such as the Process Definition Interchange Process Model Specification ([7]) and the Workflow Standard for Interoperability ([6]) to name those that are relevant for this paper. The WfMC has meanwhile developed an XML binding over HTTP of the interoperability interface ([6]) that is documented in [8]. With this specification, the WfMC introduces a protocol that allows for the exchange of messages between workflow engines across firewall systems that usually block most network traffic with exception of e.g. HTTP traffic and other dedicated network services. The protocol allows for the instantiation of workflow instances from existing workflow definitions, and the interchange of information between workflow instances during their lifecycle.

Likewise, the Object Management Group (OMG) has developed a workflow management facility standard for workflow management [18], [19], [39]. It focuses on the workflow interoperability and monitoring interfaces (interfaces 4 and 5 in the workflow reference model, [19]) and defines an interface definition in IDL that can be mapped to CORBA interfaces.

The RosettaNet consortium [37] focuses on defining standard interfaces between partners for business process integration in different industry supply chains. The standard captures various business processes and specifies them in terms of Partner Interface Processes (PIPs). The PIPs define the processes and data elements necessary for a broad set of supply chain scenarios. The PIPs only define the *interface* tasks that supply chain partners commonly participate in, but not the internal, local processes used by any partner to carry out businesses. It is the responsibility of each partner to identify how its internal processes and systems align to the PIPs. The PIP approach does address the issue of inter-enterprise process integration for enabling plug-and-play for new partners into the supply chain. This requires all partners to comply to the PIP in use. The PIP specifications focus primarily on architecting the information to be exchanged at the connection points of partner business processes; they do not focus on a common process-level specification for all the partners. The PIP do not offer a model of execution; for instance, it does not intend to specify how the partner process instances are synchronised, or made to be aware of the progress of the peer processes. Examples of PIPs include product and technical data interchange, management of purchase orders and purchase order status.

Further work to standardise a business process model is part of the UML profile for Enterprise Distributed Object Computing (EDOC) [24], being conducted by OMG. This work aims at identifying a minimal meta-model for the specification of business processes, irrespective of the underlying supporting technology. The objective is to separate business process specifications from the intricacies of underlying technologies that can be used to support these descriptions, be they WfMS, component platforms such as EJB, CCM, COM+, or any other implementation choice. The OMG joint submission for Enterprise Distributed Object Computing proposes business events and process events for the communication between workflows. Business events are the supertypes of process events. “Business events are state changes whose occurrence is of significance to the execution of business processes. Typically business events reflect state changes in Business Entities. These can be thought of as entity events. Examples are the approval of a Purchase Order, or Receipt of a Payment. A more indirect type of business event is a state change to a business process or to a collaboration between two business processes. These are called process events.” [20] Both, the business events and the process events are protocol-free messages.

The ebXML.org group [10], [11], seeks to provide an open XML-based infrastructure enabling the global use of electronic business in an interoperable, secure, and consistent manner by all parties with the aim to provide support specifically for the B2B process issues - currently not addressed in detail within the UML profile for

EDOC and WfMC standards. ebXML seeks to provide the technical infrastructure and message formats for XML-based cross-organisational workflows. The ebXML architecture provides: (1) A way to define business processes and their associated messages and content, (2) a way to register and discover business process sequences with related message exchanges, (3) a way to define company profiles, (4) a way to define trading partner agreements, and (5) a uniform message transport layer. The ebXML business process developments are centered on support for trading partner agreements. As a result, there is a limited description of flows between business tasks.

Workflow Modelling

Workflow Models and Business Process Models are closely interrelated, however, workflow models have a stronger focus on the underlying notion of workflow management, i.e. the automation of business processes.

Scheer et al describe in [21] an integrated concept to describe and execute business processes. The authors call it the House of Business Engineering (HOBE). The authors distinguish four main elements in their framework:

Process Design which is concerned with process and reference models, knowledge management, controlling an simulation.
Process Management which is about the continuous improvement of processes through monitoring and time and capacity management.
Process Workflow is about the execution of processes and the routing of objects.
Process Application is the technical implementation of workflows through application objects.

The proposed HOBE strongly resembles the workflow reference model, as described in [19]. However, HOBE describes workflow system functionality on a higher, logical, level.

Georgakopoulos et al [15] describe that current workflow models couple activities with their implementations, similar to a procedure call in traditional programming languages. A process designer has to implement all possible alternatives, usually in a subprocess. This solution complicates the maintenance of a business process. The problem is alleviated in virtual enterprises that use reference multi-enterprise process. The paper continues that virtual enterprises can be coupled by reference process-based multi-enterprise process (MEP) (tightly coupled) and by service-based multi-enterprise process (loosely coupled).

Reference multi-enterprise processes are agreed upon by the business partners or are best practice multi-enterprise processes, while service-based multi-enterprise processes are composed dynamically. Reference process-based multi-enterprise processes are appropriate for relatively long-lived virtual enterprises with cooperation partners that are willing to adapt their single-enterprise processes (SEP) to implement the parts of the MEP reference process they are supposed to perform. Service-based MEPs allow for greater flexibility and autonomy of the business participants. Service-based MEPs are appropriate for loosely coupled virtual enterprises, i.e. virtual enterprises consisting of enterprises that want to hide their internal SEPs and choose not to change their SEPs. Their target are short-lived virtual enterprises (one eCommerce transaction). A reference MEP is a process that consists of abstract sub processes that lack implementation. Abstract sub processes and the MEP are implemented by SEPs provided by the enterprises in the VE. All enterprises can have a top-level SEP that implements the reference MEP. Existing reference process concepts provide no solutions for implementation and enactment of reference-process-based MEPs.

A service-based MEP fuses together services provided by different enterprises. Services may encapsulate SEPs or other applications provided by non-workflow systems. From the perspective of the MEP, services are black boxes. Service activities are service proxies that represent the requirements to a service in the MEP. In service-based MEP environment, there is no agreement of the VE partners that deals with service autonomy and heterogeneity. Services provide information during execution. The paper does not describe an infrastructure to support MEPs and the availability of SEP systems to support MEP requirements.

Liu and Shen [43] describe a novel approach of modelling workflows with process views. They argue that business processes need to selectively expose activities without disclosing details about them. This concept is similar to views in database systems. The authors provide a formal model of workflow and extend it to virtual process views. A virtual process is a non-empty set of one or more virtual and non-virtual activities and their dependencies. The authors also discuss the problems of order preservation in virtual processes and algorithms to generate minimum virtual activities.

Chiu et. al. [4] utilise the concept of workflow views in a cross-organisational workflow environment. The focus of their work is on e-contracts, which utilise workflow views to abstract from them. The paper does not discuss the communication dependency issues of cross-organisational workflows or approaches on how to generate workflow views from workflows.

In [40] Preuner and Schrefl are looking at the composition of workflows in an eService environment by following an Aspect-oriented approach. The authors acknowledge that workflows need to be abstracted in order to hide internal data, when being used in a collaborative environment. The authors acknowledge the requirement of coordination services that are used in a composed workflow to achieved the desired order of execution of the involved workflows.

Industry Solutions

Industry Solutions, such as SAP Netweaver, Microsoft Biztalk, BEA Weblogic Integration, and IBM Websphere MQ Family (formerly called MQSeries) provide the functionality to model and execute cross-system communication with a business process management (BPM) approach. They support communication by means of state-of-the art message queuing and provide means to translate information between different schemas. Despite following different approaches to achieve these objectives, they require, to the best of our knowledge, that an existing process to be changed or augmented for the purpose of interaction across system boundaries, or a separate abstract process to be independently modelled and linked to its existing process, which is then required to be fragmented to achieve nested execution behaviour of the existing processes in relation to its abstract process.

In terms of cross-organisational workflow modelling, several authors meanwhile argue for a selectable and scalable exposition of private workflow tasks in a public context [43], [4], [23], [32]. Workflow views are considered to provide the required mechanisms for course (black-box), medium (grey box), and fine-grained (white box) visibility of private tasks to the outside world or a set of partners. The workflow view approach shall allow for a fine-tuning of visibility and privacy according to the desired requirements in a particular coalition. In other words the intent is to leave a private workflow unchanged and to relate it to a process-based interface which can be adapted to satisfy specific business requirements.

We need to investigate approaches on how to support the above-mentioned contradicting requirements of public visibility versus privacy and consequently how to architect execution support of workflow instances that are based on a view-based workflow model. The model itself has been formalized in [23] and is briefly summarised in this paper. It's concepts build upon a directed graph approach with adjacent matrix representation for the formal definition of transformation operators. It includes bi-directional operators that transform private workflows into workflow views (and vice versa) and operators that interweave workflow views into coalition workflows and those that break up coalition workflows into workflow views. In our model, the main computational entities are private workflows, workflow views, and coalition workflows. From a structural perspective, private workflows consist of (private) tasks and (private) dependencies, whilst workflow views consist of workflow view tasks (synonym: virtual tasks) and workflow view dependencies (synomym: virtual dependencies). This modelling concept is depicted in Figure 4.

Private workflows are only known to their owning entity, i.e., organisation. Workflow views are abstractions of these private workflows and are known to business partners that cooperate in a particular context, denominated Coalition. Workflow views are, in BPEL4WS terminology, abstract processes, which outsource their implementation to private workflows (BPEL4WS: executable workflows). $1 \dots n$ view models can abstract from a single private workflow and thus reflect the specific interaction requirements of multiple Coalitions (*coalition-adapted views of private workflows* in Figure 4). In order to specify dependencies between workflow views from different organisations we introduce the concept of a shared business process, which we call *coalition workflow*. A coalition workflow *references* workflow views, thus distributing the overall execution of the coalition workflow. My means of distribution and outsourcing, a coalition workflow indirectly connects private business processes in a cross-enterprise business scenario. Workflow views are exposed as interaction points, which can be used by coalition workflows to form a cross-organisational workflow. The inter-enterprise coordination thus builds on a distributed business process model where partners manage their own part of the

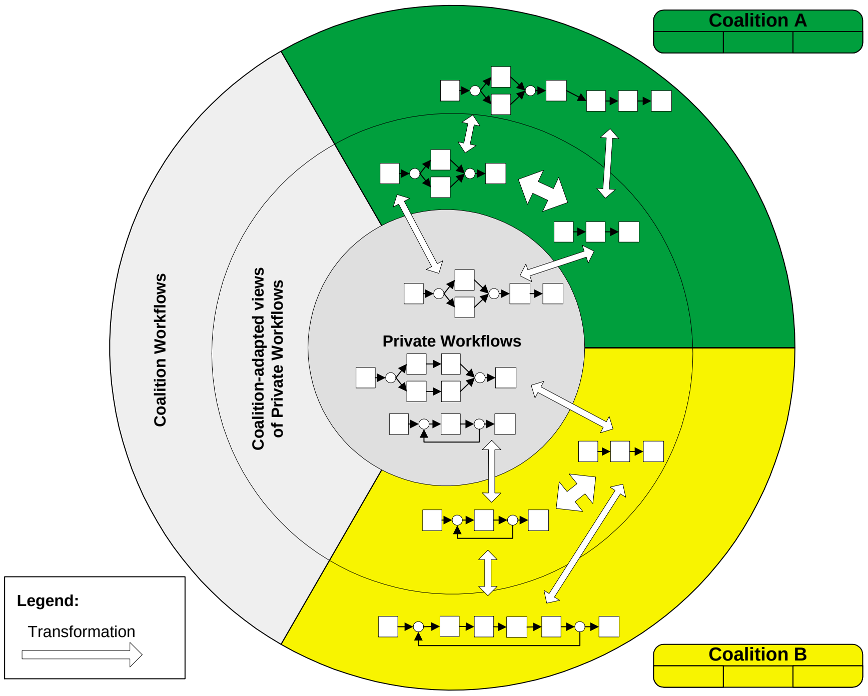


Fig. 4. Workflow View Concept: Overview

overall business process. A coalition workflow specifies tasks that each of the parties is required to perform as agreed in their contract. The tasks in the workflow views conduct conversation with their peers according to the conceptual specification in the coalition workflow. Technically, this conversation is supported by horizontal service protocols, such as SOAP and higher-level protocols, such as WS-Coordination, WS-Transaction, and (parts of) ebXML; on top of these reside vertical service protocols, such as RosettaNet and (parts of) ebXML that are able to express domain-specific conversation requirements [25].

In principle, the workflow view model can be exposed by various technical means and schemata. We have proposed an initial schema in [30]. The coalition workflow model and its constituting workflow view models can be considered as a functional description of the relationship between two or more parties. They can therefore be considered as an electronic contract. Our notion is aligned with the one of an electronic contract in Crossflow [28]. In our model we particularly express two aspects:

(1) the structural dependencies between workflow views on the level of a coalition workflow,
(2) a not publicly visible binding from workflow views to private workflows that implement the publicly-visible behaviour of each party.

An electronic contract can be instantiated by means of instances of the corresponding workflow view models.

2 Architectural Aspects

There are two fundamental architectural principles for workflow integration architecture. One is that the communication partners interact directly - we call this *unmediated* in this paper. The other is that a third entity mediates or supports the communication. This approach has several names, such as orchestration (Microsoft), choreography (SAP/Sun), brokering (OMG), or mediation. We use the term mediation in this paper. We explore characteristics of unmediated and mediated interaction in the following subsections.

2.1 Scenario

During the remainder of this paper we regularly refer to the following scenario that is depicted in Figure 5 in order to explain our concepts.

In Figure 5 we face the situation of two workflow management systems that are supposed to interact. In the directed graph representation, a rectangle represents an activity task, a circle represents a routing task, and an arrow represents a dependency between two tasks. For example, routing task m_3 is called after activity task m_2 has finished.

Each of the workflow management systems generates a suitable view for the desired interaction, which is based on the information that should be hidden and the required tasks that the business partners need to see in order to be able to interact. We envisage that this will often be a task with human involvement, but which can be greatly supported by software, which proposes syntactically correct generalisations based on our definitions of a workflow in [23].

In our scenario, we assume that there are six view-tasks, denoted as $\{l_1, l_2, l_3, p_1, p_2, p_3\}$, and one rule that describe how they are expected to interact. The rule is expressed through a coalition workflow M with its tasks $\{m_1, m_2, m_3, m_4, m_5, m_6, m_7, m_8\}$. In the directed graph representation in Figure 5 each activity task in M refers in brackets its corresponding view task. M is a logical representation of this interaction. However, M is not executed, but realised through P and L, which are implemented by O and K respectively.

For the following considerations in this paper we postulate that private workflows must remain unchanged. We argue that is a core requirement because private workflows represent best practice and a considerable intellectual property.

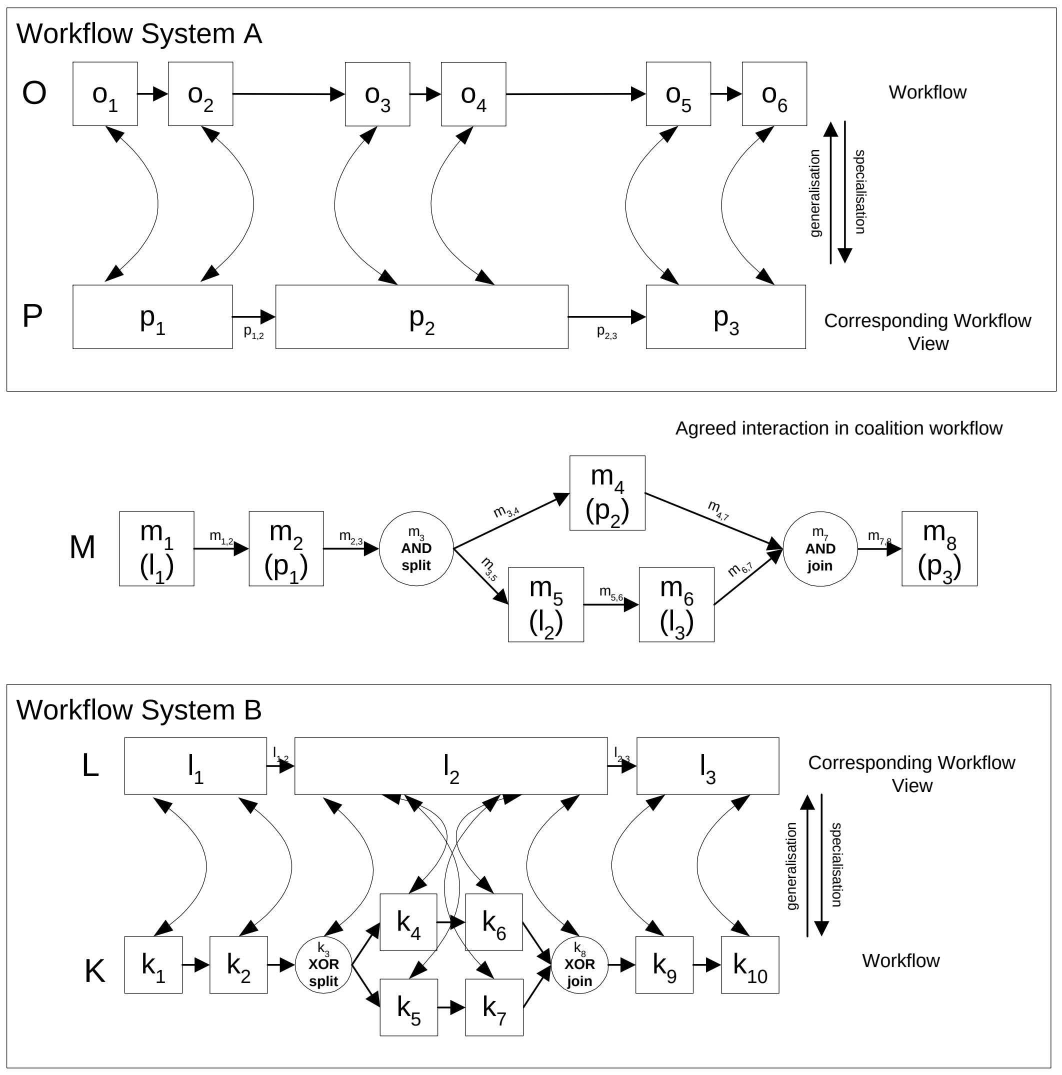


Fig. 5. Scenario

2.2 Unmediated Communication

We consider mutual awareness between the communication partners. All communication is direct between them. This requires that the they agree on a method and protocols of interaction. There are up to $n^2 - n$ individual communication paths amongst n participants.

2.3 Mediated Communication

In a mediated environment, there is one central participant that is able to route information to the communication partners, which do not have to know each other. Prior knowledge about the mediator is sufficient. All or some communication is routed through the mediator. This decouples the sender and receiver of information and reduces the number of individual communication paths to 2*n* where *n* is the number of participants.

A mediator is an entity that coordinates interactions. The notion of a mediator for distributed workflows was first introduced in the Vega project in 1997. It was finally specified in the document *Implementation of the Distributed Workflow Service*, [42], and further referred to in *Demonstration of Virtual LSE enterprise - Vega 5th Review Demonstration*, [45], and in *Public Final Report*, [46]. It was referred to under the terms of a *Workflow Backbone* and *Distributed Workflow Service*. The idea of coordinating distributed workflows has since received a greater academic and industrial attention, such as by Intel in [44]. The authors propose a Process Coordination Framework (PCF) that provides the building blocks required for Web services enabled e-business automation. They distinguish between private processes and public collaborative processes. Mejia et al explore in [27] the concept of a *Virtual Enterprise Broker (VEB)* to perform pre-defined core processes of virtual enterprises: (i) Analysis of market requirements, (ii) Project planning, (iii) Project Execution, and (iv) Customer service. Other authors consider information brokers for information retrieval and assignment to virtual enterprise partners. Examples of those research activities are described in [9] and [22]. The idea of a coordinating entity in collaborations seems to be attractive, as it can be trusted and knowledgeable about partner interactions, which it can support. Mediation has at least two facets:

Stateless. The mediation passes messages from sender to receiver.

Stateful. By stateful mediation we distinguish:

- **passive** This model includes the characteristics of stateless mediation, but also logs the interaction in a persistent storage, therefore facilitating monitoring and error handling. Stateful passive mediation allows to match request and respond messages and to assign them to the right participant. This indirect syle of communication is particularly applicable in scenarios where the trading parties must not know about each other, because the existence of an order placed by one party would express the shortage of a good which could be used by the recipient of the order to compete with the order provider in a regional market. This has been observed in the Oil & Gas domain ([38]).
- **active** This model is increasingly popular by enterprise integration technology providers. In the model, a central workflow management system schedules tasks to the involved partners. The model includes stateful passive mediation, but also actively drives the partner's interaction by executing a central coalition workflow and invoking the communication partner's IT systems to perform their work. In the following we further elaborate on this model and explain our particular choice for the *hybrid* model of communication that operates without a central workflow management system.

2.3.1 *On Active Stateful Mediation*

Figure 6 depicts a situation where a central workflow engine mediates the interaction of the partner's workflow systems. The figure shows the control-flow dependencies between the partners workflow views and the coalition workflow.

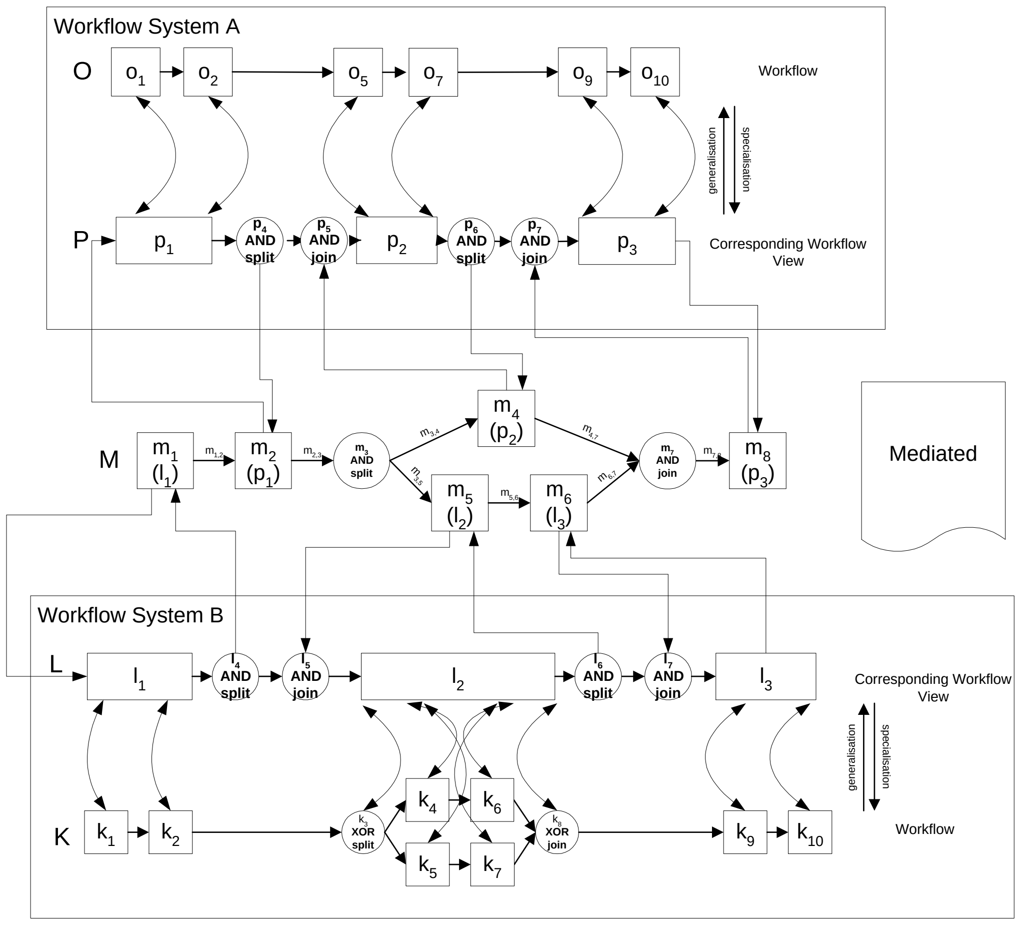


Fig. 6. Active Stateful Mediation

In this situation, the coalition workflow *M* is being physically instantiated in a workflow engine (state machine). This engine interacts with other workflow systems A and B to execute workflows *P* and *L*. To achieve this it is necessary to embed each task in *P* and *L* into a set of split and join routing tasks in order to stop *P* and *L* to execute on their own without involvement of *M*, we also say that *M* binds *P* and *L* very tightly. We argue, however, that there is no real need for a central state machine to run a coalition workflow, because there are already 2 (in general *n*) individual state machines that execute their respective private workflows. This situation is different though, when tasks of the coalition workflow are not implemented through a workflow view. We encountered this situation often in electronic marketplaces and private exchanges. For ease of setup, the marketplace operators sometimes prefer to use their central workflow system to interact with their customers, whilst scheduling and coordinating customer verification and billing services. Conceptually, this would be realised through a marketplace workflow system that implements the marketplace's workflows. This setup has the advantage that it facilitates monitoring: the coalition workflow reflects the states of the involved partner workflows. A monitoring tool can directly display the coalition's workflow status information. There is no need to collect monitoring data from the involved workflow views. However, we argue that if monitoring is the main motivation for stateful mediation, then this is better achieved through stateful passive mediation which includes a central monitoring service. However, the model is applicable in scenarios that do not involve sufficiently functional workflow management systems on the partners sides.

2.3.2 Hybrid

In a hybrid approach, stateless and stateful passive mediation services are used where required by the communication partners. Mediation is being used for monitoring, persistence of messages, and for offline-support, whilst other information is exchanged directly been the communication partners. In our model we assume that the participating workflow management systems are sufficiently functional to actively drive their own private workflows, workflow views, and send messages to each other. We believe that mediation services are useful to establish the communication between workflow management systems. However once the systems have been bound to each other and agreement has been reached in which format data is to be exchanged, the mediation would not necessarily need to continue to support the interaction other than providing a communication infrastructure, such as the Internet. Therefore we build on the hybrid concept for in our architectural model that we discuss in Section 4.

3 On Stateful Interaction

In this section we further investigate into communication aspects between the entities of our cross-organisational workflow model. We analyse how private workflow can statefully interact with workflow view and how workflow views can statefully interact with each other, either directly or through a mediator.

3.1 Control Flow Dependencies

Control Flow dependencies express how two or more workflows can interact through the introduction of synchronizing tasks and dependencies. A dependency *d* defines the execution dependency between two objects in a workflow model. By connecting tasks through dependencies, we build the structure of a workflow specification where dependencies represent the edges of an adjacent digraph whilst tasks represent vertices. Route tasks, such as joins and splits coordinate the control flows of the involved workflows to ensure order preservation of the overall workflow that results from the interaction of the individual workflows ([23]). Control flow dependencies provide a loose coupling between workflows, because they merely pass state and workflow-relevant data from one workflow to another.

With regards to our example above the concept of control flow dependencies requires to augment workflow views through synchronising tasks $\{i_4, i_5\}$ and $\{p_4, p_5\}$ (see the circles in *I* and *P*) in order to preserve the order of execution. This is illustrated in Figure 7.

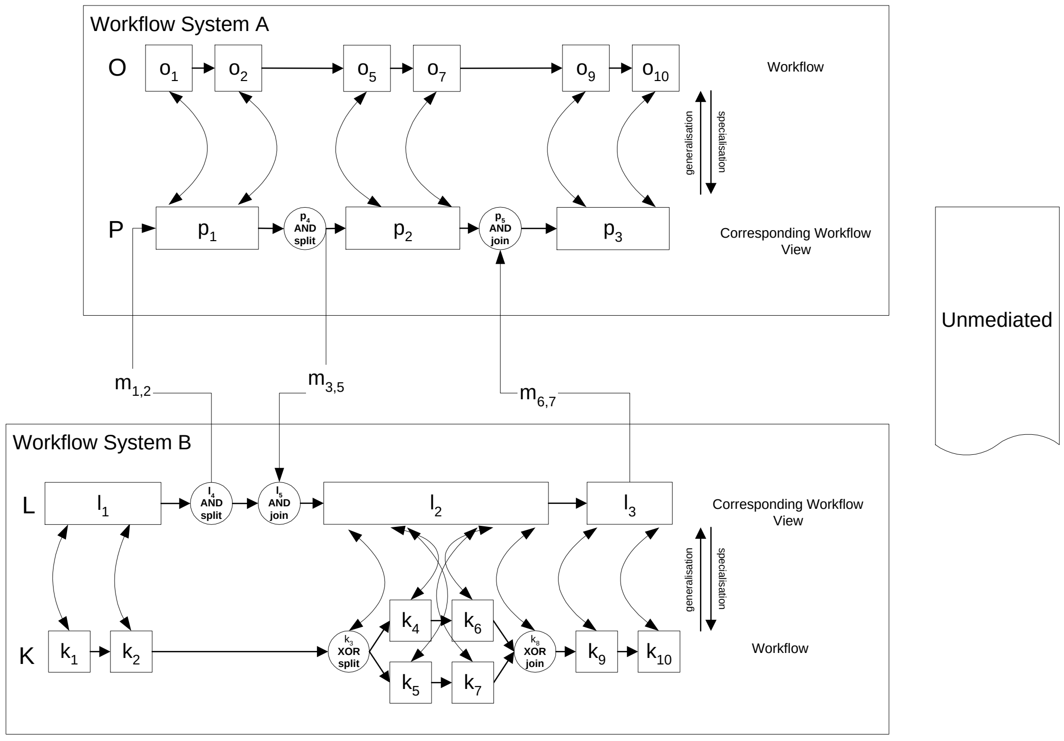


Fig. 7. Control Flow Dependencies

3.2 State dependencies

With respect to the introduction of outsourced workflows in Section 1.2.2 one can consider a workflow view as a proxy for its corresponding private workflow. In other words, a workflow view is outsourcing its implementation to its corresponding private workflow. There is a very tight coupling required between them such that the workflow view always accurately represents the state of its corresponding private workflow in regular and exceptional circumstances and that any valid messages that are sent to the workflow view by a third entity are forwarded to the appropriate task in the corresponding private workflow.

There is an obvious relationship between the execution states of a workflow view task to its aggregated private tasks. In state-of-the-art workflow management, the states, that a task can obtain are [7]:

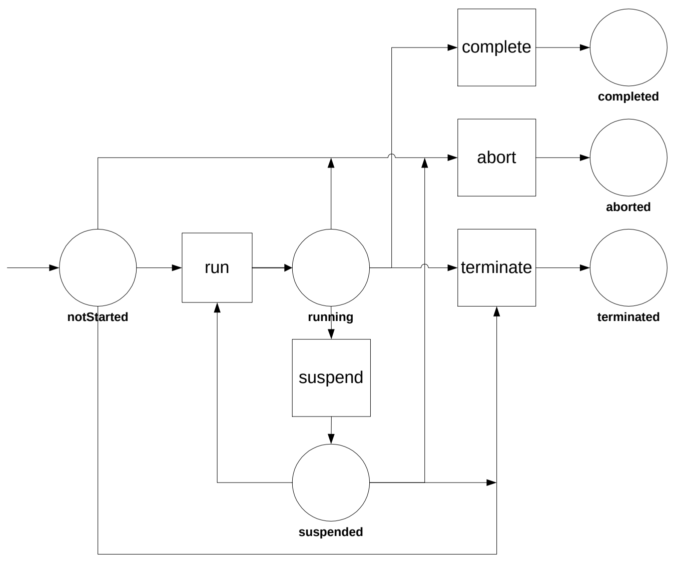


Fig. 8. Task States and Transition Diagram (analogous to [7]) (Petri Net Representation)

open: the task is being enacted
open.running: the task is executing
open.notRunning: the task is temporarily not executing
open.notRunning.notStarted: the task has been created, but was not started yet
open.notRunning.suspended: execution of the task was temporarily suspended
closed: enactment of the task has been finished
closed.aborted: enactment of the task has been aborted by a user
closed.terminated: enactment of the task has been terminated by a user
closed.completed: enactment of the task has completed normally

We illustrate the states that occur within a task, view task respectively, in Figure 8, with a Petri-Net representation. This representation is therefore more fine-grained than a workflow model, which is on the level of tasks. In the figure, a circle represents a state, whilst a square represents an event. The semantics is such that when the system is in a state s_0 only events associated with output transitions of s_0 can change the state of the system.

We now consider a workflow instance K and its corresponding view instance L. Let k be a task instance in K and l be a task instance in L. State dependencies express that k and l must not change their state unless the state change satisfies rules set forth that describe how states in K and L relate. Let change-state, cs, be a function of task t, which requests t to change from its current state s_0 into a new state s_1. We denote this through: $\text{cs}(s_1)\ t$. We denote the state transition from s_0 to s_1 as $s_0 \longrightarrow s_1$.

Once l requests: $\text{cs}(s_1)\ k$, k performs the following assessment:

(1) Is the state transition $s_0 \rightarrow s_1$ valid, i.e., is it reflected by the adjacent state

transition model of *k*? If false, return false. If true continue with step 2.

(2) Are all operational resources available? I.e., are all required private dependencies active, or, when *k* is the first task in K, is the local workflow engine ready to instantiate K? If false, return false. If true, continue with step 3.

(3) Perform state transition $s_0 \rightarrow s_1$. s_1 is now the state of *k*. Return true.

Similarly, *l* would perform the following assessment once *k* requests cs(s_1) *l*:

(1) Is the state transition $s_0 \rightarrow s_1$ valid? If false, return false. If true continue with step 2.

(2) Are all operational resources available? I.e., are all required workflow view dependencies active, unless *l* is the first task in L. If false, return false. If true, continue with step 3.

(3) Perform state transition $s_0 \rightarrow s_1$. s_1 is now the state of *l*. Return true.

We now explore approaches to achieve a flexible relationship between states of workflow view task instance and its corresponding private task instances.

3.2.1 State Mapping

One approach is to map between states of task instances to the state of the corresponding workflow view task instance and vice versa as shown in Table 1.

Workflow View Task Instance	Task Instances
open.running	one *open.running* AND none (*closed.aborted* OR *closed.terminated*)
open.notRunning	one *open.notRunning* AND none *closed*
open.notRunning.notStarted	first task *open.notRunning.notStarted*
open.notRunning.suspended	one *open.notRunning.suspended* AND none (*open.running* OR *closed.aborted* OR *closed.terminated*)
closed.aborted	one *closed.aborted*
closed.terminated	one *closed.terminated*
closed.completed	last task AND all other tasks that have been instantiated have been *closed.completed*

Table 1
State Propagation from Task Instances to Workflow View Task Instance

This approach is unable to capture the semantics and the structure of the private workflow instance. Consider the parallel execution of two task instances and one of them aborts. This does not automatically mean that the corresponding workflow view task instance has to abort as well. This is in contradiction to when a workflow view task instance receives a request to enter into a particular state, particularly *open.notRunning.suspended*, *closed.aborted*, and *closed.terminated*. In this case, all corresponding private task instances have to enter the respective states of: *open.notRunning.suspended*, *closed.aborted*, and *closed.terminated* according to the individual state transition model.

3.2.2 Explicit Modelling

In another approach, a workflow model would be rich enough to explicitly and individually model the relationships between workflow view task instance and private task instance states, whilst providing a default in the case of absence of an explicit correlation as suggested in Table 1. We argue that this approach has the advantage to be flexible whilst pragmatic. It would also allow deriving state-dependencies from a workflow model that describe the relationships between tasks and their workflow view task. Such a model needs to be rich enough to capture the communication from the private workflow to its corresponding view and vice versa, taking into account that any messages from the view originate from the coalition.

We consider the following example:

Let K be a workflow with tasks k_1 and k_2 that form a path. Let L be a workflow view with the workflow view task l_1 that represents k_1 and k_2. Figure 9 illustrates the state transition diagram of K, whilst Figure 10 shows L. In both figures, we again use a Petri-Net notation.

We assume that once task instance k_1 or k_2 enters into a state, it notifies l_1 and vice versa. We also assume that in L each state has a corresponding tentative state, denoted as *tstate*. These tentative states allow l_1 to revert to the original state in which it was before it received an event, in case k_1 and/or k_2 were unable to execute the state-change-request. This has been highlighted in Figure 10 as light-grey shading for the state *completed*. We assume in our example that k_1 and l_1 are in the initial state *notStarted*. In both figures the events that originate from the coalition are prefixed with c, e.g. cRun, whilst events that stem from l_1 are prefixed with l_1 once they are referred to in k_1 and k_2. Events from k_1 and k_2 are prefixed with k_1, k_2 respectively once they are referred to in l_1. Events without any prefix originate from the respective entity where they are used.

If l_1 receives the event cRun from the coalition, it enters the tentative state *tRunning* and passes on the event $\mathrm{l1}_{\mathrm{tRun}}$ to k_1. k_1 then enters the state *running* and sends event $\mathrm{k1}_{\mathrm{run}}$ to l_1. l_1 then enters the state *running*.

If k_1 sends a noCommit event, indicating that it was not able to perform the request, then l_1 returns to its original state.

If K is instantiated without coalition request by its own workflow engine, then the following exchange of events takes place: k_1 enters the state *running* through

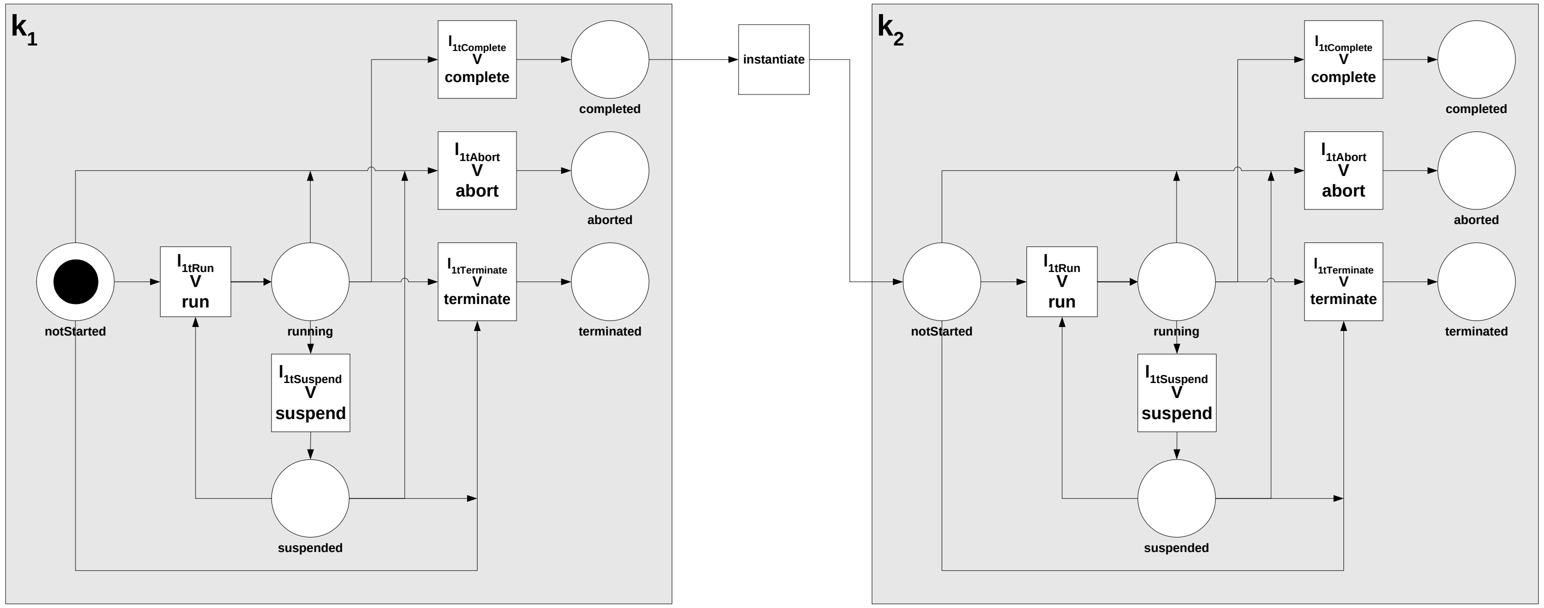


Fig. 9. State Transition Diagram of K (Petri Net Representation)

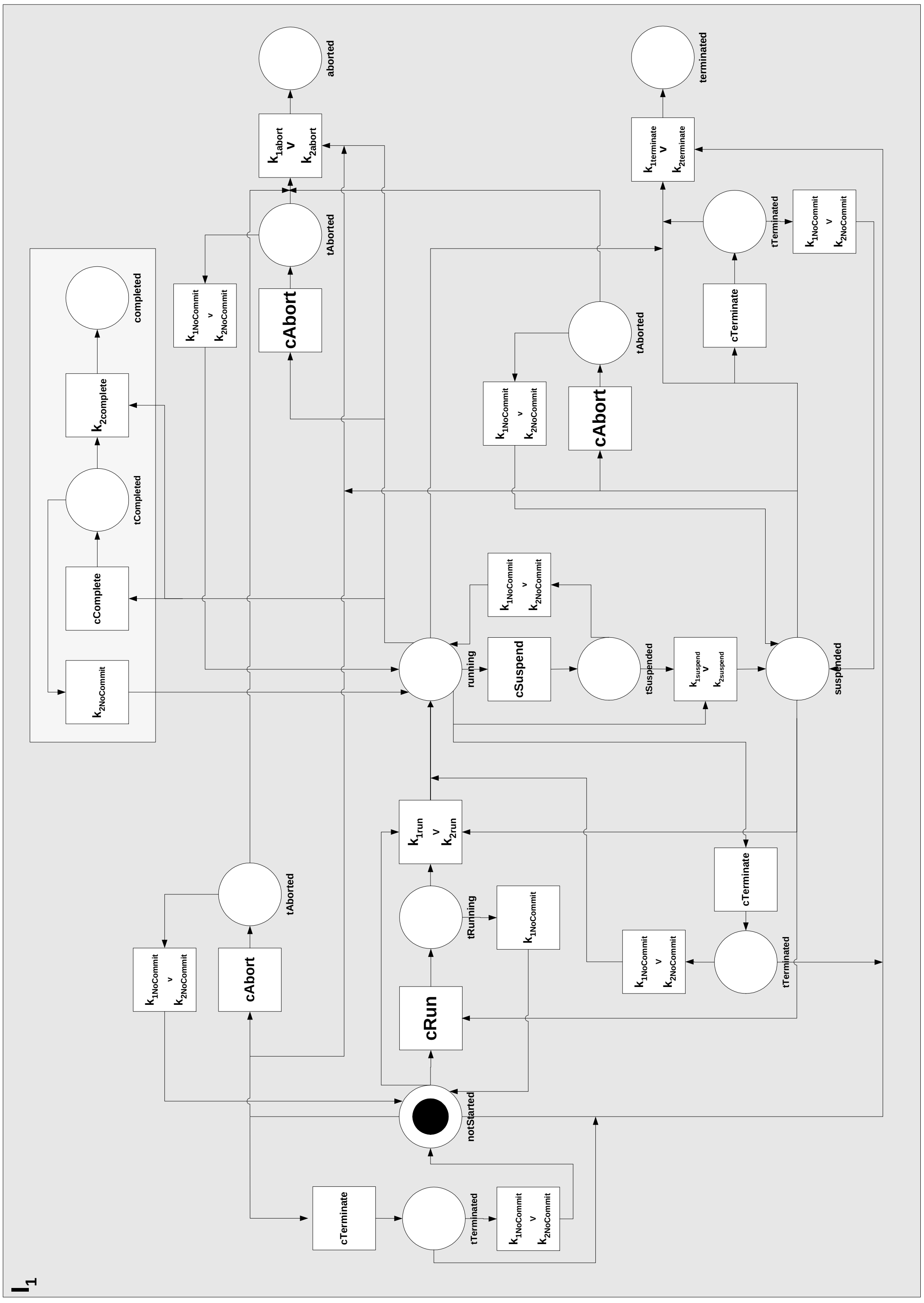


Fig. 10. State Transition Diagram of L (Petri Net Representation)

a request of its workflow engine and sends event $k_{1_{run}}$ to l_1. l_1 then enters state *running*.

We observe, that the states in K are of higher order than the states in L. The rationale is that K is a workflow that is implemented by a system, whilst L is a mere abstraction of K and is implemented through K.

The general structure of the Petri-Net of l_1 would remain unchanged in the situation where k_1 and k_2 execute in parallel. However, the events from the private tasks would be correlated differently. For example, l_1 could only complete if both k_1 and k_2 completed.

3.3 Summary

As control flow dependencies are widely supported by today's workflow management systems we propose to utilise them to interconnect augmented workflow views. We further propose to utilise state-dependencies to connect private workflow to its corresponding workflow view to achieve their tight integration. We argue not to use state-dependencies to connect workflow views for three main reasons:

(1) We cannot assume that the involved workflow management systems support the same internal states.
(2) This approach would make the relationships between workflow views opaque.
(3) This would interconnect all states of view-tasks and would lead to a very tight coupling, which would contradict the requirement of autonomy in a cross-organisational workflow environment. Control flow dependencies connect the closed-state of a preceding task to the open-state of the following tasks. Utilising control-flow dependencies to connect workflow view and private workflow would not allow an accurate and timely interchange of state changes.

In Sections 4 and 5 we discuss an architecture and an extended workflow management system that materialise our proposed communication model.

4 Cross-Organisational Workflow Architecture

To demonstrate the context in which the distributed workflow model (Section 1.4) and the communication paradigms (Section 3) between the entities of such model would be applied, we propose a distributed workflow architecture to support cross-organisational workflows. This architecture embraces the distributed process model and therefore allows for business interactions between organisations without requiring them to have full knowledge of the structure of each other's processes and without expositing confidential data to their partners.

The Cross-Organisational Workflow Architecture (CWA) that we describe in this section is based on the notion of hybrid interaction that we discussed in Section 2.3. This enables cross-organisational workflow scenarios in which the business partners don't want to reveal their identity. We argue that this architecture can support non-mediated interactions by removing the mediator, connecting the gateways of the workflow management systems, and distributing the information of the mediator's repositories to the involved workflow management systems. The proposed architecture extends a company's existing workflow infrastructures and enables process-oriented communication with their partners and customers.

Figure 11 depicts two workflow management systems, a mediator and a certificate authority (CA). The CA is outside the scope of our consideration and has therefore been depicted as a black box. However, it has been included in the architecture to reflect on PKI security aspects. The workflow management systems and the mediator are constituted by a set of building blocks, which we introduce next in more detail.

4.1 Architectural Entities

In this section we discuss the architectural entities of the proposed extended workflow management system and the mediator for cross-organisational workflows. The architecture of the workflow management system is related to the standard workflow reference model proposed by the WfMC [5] and OMG [19] but has been extended to support our proposed cross-organisational workflow model. The extensions are through an extended workflow engine, a dedicated workflow view modeller, a gateway, and a security manager, whilst the mediator is not covered by any workflow standards approach yet.

Private Workflow Modeller

This entity is element of the extended workflow management system. It is a state-of-the art modeller for private workflows and is only mentioned for completeness of the discussion. It supports the lifecycle of private workflow models. The modeller stores its workflow models in the private workflow & workflow view repository.

View Modeller

This entity is part of the extended workflow management system. It allows for bottom-up and top-down modelling of private workflows via workflow views into coalition workflow, and vice versa the decomposition of existing coalition workflows into workflow views which are implemented by private workflows. Through interaction with a user and coalition knowledge, it supports:

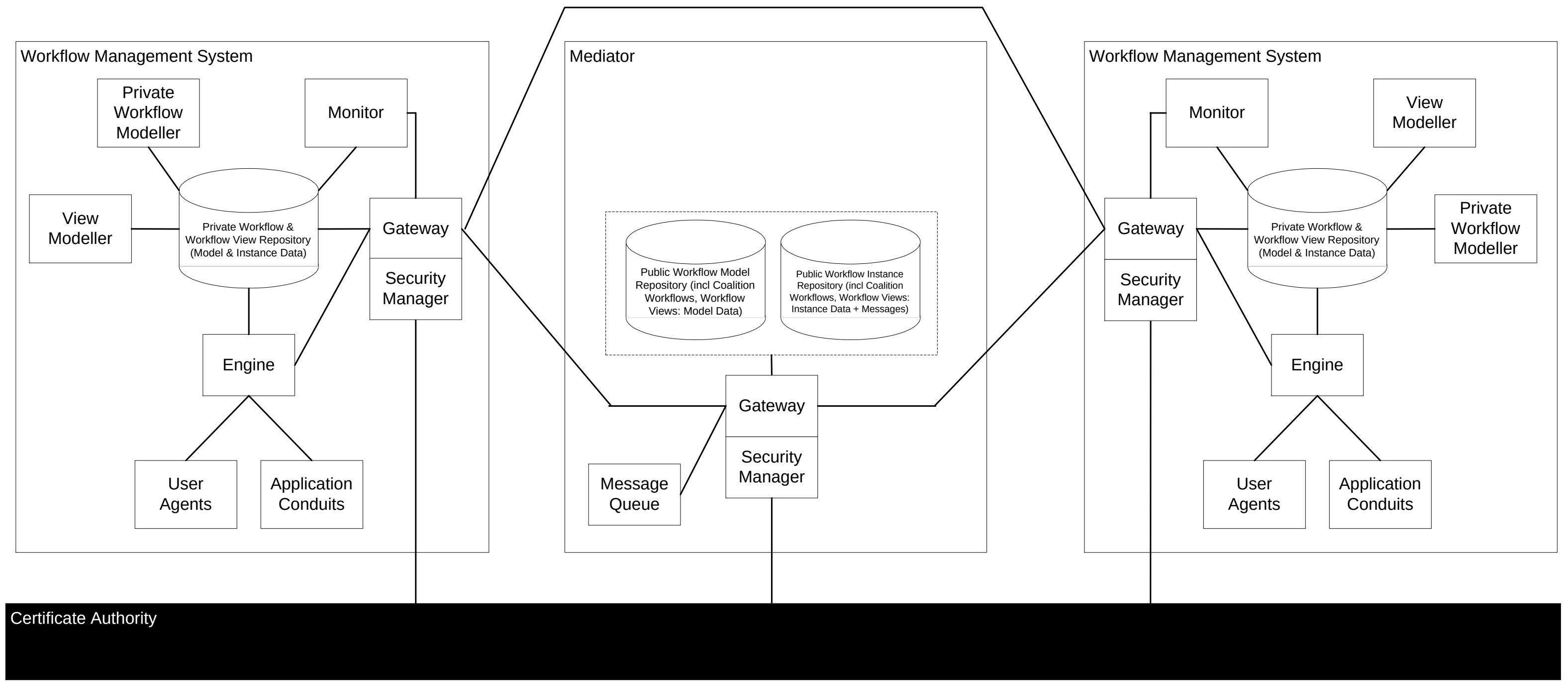

Workflow Management System
Private Workflow Modeller
Monitor
View Modeller
Private Workflow & Workflow View Repository (Model & Instance Data)
Gateway
Security Manager
Engine
User Agents
Application Conduits
Mediator
Public Workflow Model Repository (incl Coalition Workflows, Workflow Views: Model Data)
Public Workflow Instance Repository (incl Coalition Workflows, Workflow Views: Instance Data + Messages)
Gateway
Security Manager
Message Queue
Workflow Management System
Monitor
View Modeller
Private Workflow & Workflow View Repository (Model & Instance Data)
Private Workflow Modeller
Gateway
Security Manager
Engine
User Agents
Application Conduits
Certificate Authority

Fig. 11. Cross-Organisational Workflow Architecture

- specialisation of workflow views into private workflows,
- generalisation of private workflow into workflow views,
- aggregation of workflow views into coalition workflows, and
- reduction of coalition workflows into workflow views.

The modeller stores workflow view models and their interdependencies to private workflows in the private workflow & workflow view repository. The coalition workflow models references workflow view models and the dependencies between them. It is therefore sufficient to store these *Coalition dependencies* in the *Public Workflow Model Repository*. Therefore, the coalition workflow is not owned by any of the participating parties. In a non-mediated execution context, each of the parties stores the receiver of outgoing coalition dependencies and the originator of incoming coalition dependencies in the participants' local entities of the *Public Workflow Instance Repositories*.

Monitor

The workflow monitor is an entity of the extended workflow management system. It tracks the execution of private workflow instances and workflow view instances. Monitoring is performed on all layers of the workflow model, i.e., coalition layer, view layer, and private layer. The availability of monitoring functionality for a particular layer is determined by the availability of instance information from the performing workflow engine. This means that a workflow monitor that accesses coalition workflow and workflow view instance data from the mediator can track workflows on the publicly visible layers, whilst a workflow monitor with additional access to private workflow instance data can track a company's respective private workflow instances.

In a mediated environment, the workflow monitor is able to query state information about other workflow view instances from the mediator's instance repository. In [41] we have discussed workflow monitoring building on this approach and developed a cross-organisational workflow monitor that is capable of monitoring coalition workflows. In a non-mediated environment, the workflow monitor is required to collect workflow view instance data from the partners' workflow management systems in order to represent the state of coalition workflow instance.

User Agents

User Agent is an entity of the extended workflow management system and is the human user's interface to the workflow system. A user agent can be a task list. It is assumed that users always belong to an organisational entity, which is represented through a workflow management system in a coalition. Therefore, the user agent is not part of the mediator, as the mediator is not representing an organisational entity and is therefore not expected to directly interact with users.

Application Conduits

Application conduits are part of the extended workflow management system and correspond to state-of-the-art application handlers. They enable the workflow management system to interface with external applications, such as back-office applications like customer relationship management systems, supply chain management, vending machines, but also front-office applications, such as a word processor or spreadsheets to mention a few examples. Application Conduits extend the reach of workflow management systems to collect and disseminate relevant data and to interact with their application environment.

Private Workflow & Workflow View Repository

This is an entity of the extended workflow management system and is an extension to state-of-the-art workflow repositories. It is able to accommodate workflow view models and instances and the relationships between private workflow and workflow view models and instances.

Workflow Engine

Workflow Engine is an entity of the extended workflow management system. It extends state-of-the art workflow engines. It is able to execute private workflows and map workflow view states to private workflow states as described in Section 3 and workflow data respectively (Section 6). The engine provides relevant information and allows access to workflow views, such that the views can interact with entities outside of the workflow management system through a gateway as described below. This means that, although the engine executes private workflows internally, it also virtually executes workflow views and ensures the appropriate interdependencies between a private workflow and its corresponding workflow view.

Gateway

This component is part of the extended workflow engine. It is a company's process interface to the outside world. It redirects all ingoing and outgoing process requests serving as a proxy. It hides internal systems from the outer world. Gateways are a frequently used approach to linking different types of components in heterogeneous environments ([36]). Some of a gateway's most important characteristics are:

Confidentiality and access protection: A gateway protects internal systems of an organisation from direct access from the outside, serving as a 'firewall'. Making an API of a system directly accessible to the outside may permit undesired activities.

Transparency/Routing: Gateways make the communication of systems transparent. Neither does a workflow designer need to know details about an organisation behind the offered service, nor does a performer in a workflow instance need to be concerned about the correct routing of message objects to the correct systems.

Format conversion: Gateways have the ability to convert outgoing message objects from the format used by the internal system to a format that can be understood by the external recipient. Likewise, incoming message objects may have to be converted so that an internal system can handle the information.

Tracking: Using this functionality, the necessary information for monitoring and cross-organisational service billing is supplied. A gateway can have a full log of all services that have been invoked on the systems of an organisation. To allow partner companies a dedicated insight into executed services, subsets of the tracking information can be supplied to them.

Error prevention and handling: Gateways can deal with failures in the interactions of systems.

Security Manager

The Security Manager is an entity of both the extended workflow engine and the mediator. It handles all security-related aspects of communication. It decrypts incoming messages and verifies the sender's identity in cooperation with a Certificate Authority. In a public-key approach, it encrypts outgoing messages with the recipient's public key.

Public Workflow Model Repository

The Public Workflow Model Repository is an entity of the mediator. It manages coalition workflow models and their corresponding workflow view models. It further stores coalition dependencies, which are those dependencies that interconnect workflow view models. Further, but beyond the scope of the considerations in this paper, the repository would store or reference required UDDI and WSDL documents required to query, bind, and invoke the systems of the communication partners ([30], [47], [50]).

Public Workflow Instance Repository

The Public Workflow Instance Repository is an entity of the mediator. It persistently stores information about the current state of execution of coalition workflows. This is for the purpose of monitoring and exception handling, e.g. when one of the communication partners has not received a message and the message needs to be re-sent.

Message Queue

The message queue is an entity of the mediator and acts in close cooperation with the public workflow instance repository. It stores messages for communication partners, therefore supporting offline scenarios. It can be considered as being similar to a mailbox. The architecture does not reference or require a particular communication protocol or technology. It does, however, require a protocol that is sufficiently expressive to allow for the modelling of messages that result in creation of workflow instances from existing workflow models and support interaction of workflow instances during their lifecycle. These core requirements are for example being supported by the Workflow Management Coalition's workflow XML specification [8].

4.2 Runtime Behaviour and Application

To investigate how the proposed architecture behaves at runtime and how to apply it we consider two aspects: Firstly, we describe how collaborative modelling is supported. Secondly, we take a look at how the architecture executes and monitors cross-organisational workflow instances.

4.2.1 Collaborative Modelling

We envisage that collaborative modelling would require top-down and bottom-up approaches. In the top-down approach, the partners agree on the interaction they want to automate and model it in a coalition workflow. Partners choose those tasks of the cross-organisational workflow that they want to implement by their private systems. Each partner applies the method of reduction to the coalition workflow in order to understand the required relationships of the partner's view tasks in the context of the coalition. Partners then specialise their workflow views by linking private workflows to them.

In the bottom-up approach, partners develop new private workflows or use existing ones and abstract them into workflow views through generalisation. Once each partner has built their respective workflow views, they apply the method of expansion on the basis of the coalition workflow definition, in order to add the required synchronizing tasks (AND-splits and AND-joins) to their workflow views. These modifications are then propagated back to the view's definition in the *private workflow & workflow view repository*.

4.2.2 Collaborative Execution

The instantiation of a cross-organisational workflow is triggered by either an external event to a workflow view or by the request of one of the partners to their private workflow management system. Eventually, one private engine starts executing a private workflow. Once it comes to a point that communication with a partner is required, the gateway requests the security manager to encrypt the message and send it off to the mediator or to the recipient. The information about possible recipients is read from the *public workflow model repository*. A coalition workflow instance is being created in the *public workflow instance repository*. In all cases, but particularly in those in which multiple recipients provide the requested service, a selection choice is being made and stored in the *public workflow instance repository*. This facilitates future communication with the right communication partner in subsequent calls and helps to verify valid origins of messages that are being exchanged throughout the lifecycle of the coalition workflow instance. Throughout the execution, the state dependencies between workflow view and corresponding private workflow are updated when changes to states occur.

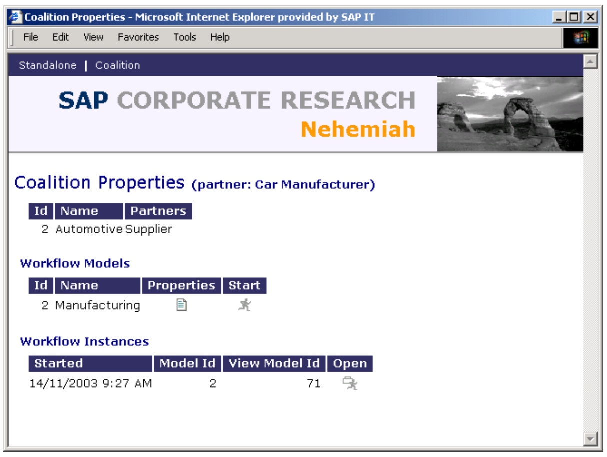
In a mediated environment, the mediator decrypts the message with the mediator's private key, encrypts it with the recipient's public key, store it and put it in a message queue and inform the appropriate participant about the new message. Once the recipient has pulled the message from the message queue, it is being removed from it. The recipient decrypts the message with their own private key and sends the message to the engine. The engine takes appropriate action by starting a new private workflow or forwarding the message to an already running instance.

In this context it is necessary to discuss how the communication partners are able to identify already instantiated workflow views in a target workflow management system. We propose a token to be passed along the communication chain that identifies the instance and the type of a coalition workflow, as per the discussions in [30]. The involved workflow management systems are able to instantiate their private workflows objects and assign them to workflow view objects that participate in the coalition workflow. The coalition workflow instance identifier is assigned to these objects. This mechanism allows us to realise not only *on own behalf*, but also *delegated* communications as per our discussions in [31].

With respect to the interaction between private workflows and their corresponding workflow views we assume that they are part of the same system solution and therefore propose that each workflow view references a private workflow and vice versa. This allows to interchange state changes as postulated in Section 3.

5 Example of a View-Based Workflow Management System

In this section we consider the prototypical implementation of a cross-organisational workflow management system which is supported by the architecture that we introduced in Section 4. We have chosen a manufacturing scenario that involves two partners *Manufacturer* and *Supplier*. We demonstrate the relationship of their respective private workflows to their adjacent workflow views, the joint coalition


Coalition Properties - Microsoft Internet Explorer provided by SAP IT

File Edit View Favorites Tools Help

Standalone | Coalition

SAP CORPORATE RESEARCH
Nehemiah

Coalition Properties (partner: Car Manufacturer)

Id	Name	Partners
2	Automotive	Supplier

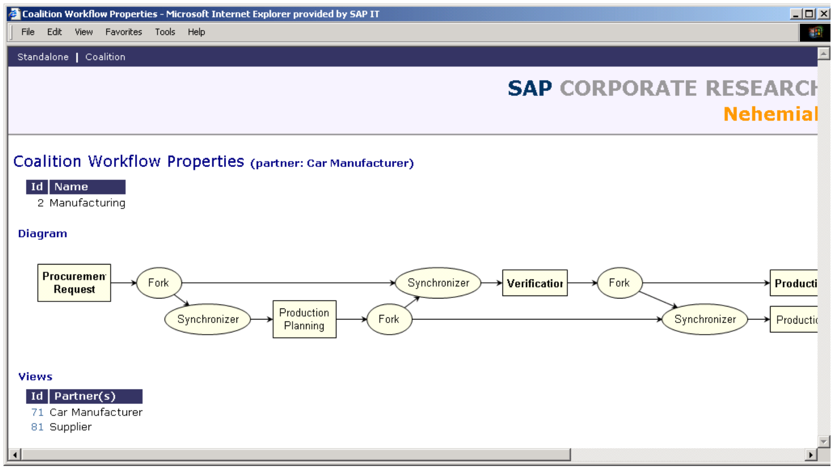
Workflow Models

Id	Name	Properties	Start
2	Manufacturing		

Workflow Instances

Started	Model Id	View Model Id	Open
14/11/2003 9:27 AM	2	71	

Fig. 12. Coalition Model Properties


Coalition Workflow Properties - Microsoft Internet Explorer provided by SAP IT

File Edit View Favorites Tools Help

Standalone | Coalition

SAP CORPORATE RESEARCH
Nehemiah

Coalition Workflow Properties (partner: Car Manufacturer)

Id	Name
2	Manufacturing

Diagram

Views

Id	Partner(s)
71	Car Manufacturer
81	Supplier

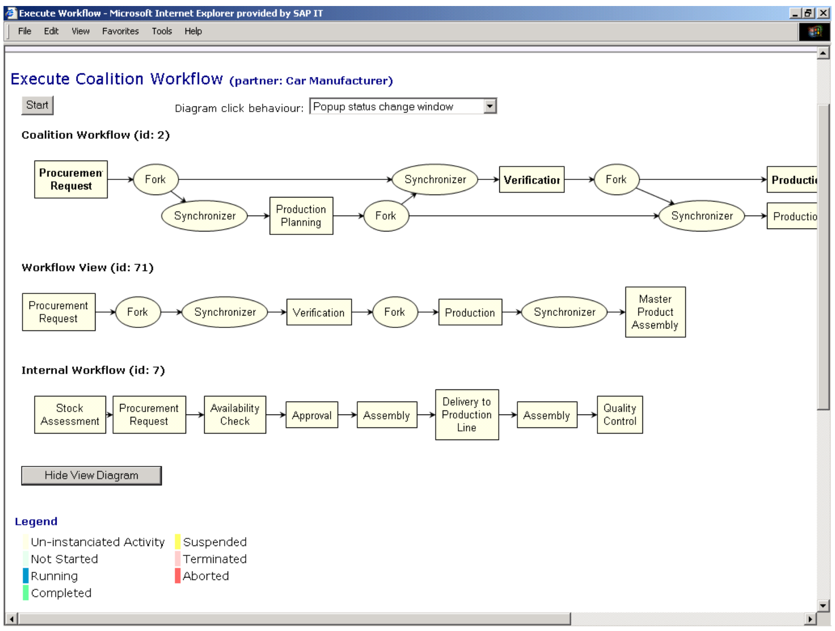
Fig. 13. Coalition Workflow Model Properties

workflow and how two workflow engines interact to execute the private workflows.

Figure 12 displays the properties of the coalition *Automotive* from the manufacturer's perspective. It shows the involved partner(s) in addition to manufacturer, i.e., supplier. Amongst the section *Workflow Models*, the underlying coalition workflow model can be viewed, which we discuss in Figure 13. Further, running instances are referred to. In the particular case, there is one running instance.

Figure 13 shows the coalition workflow as agreed by the manufacturer and the supplier. The directed graph in the figure has been drawn from the manufacturer's perspective, i.e., the tasks that would be performed by the manufacturer are highlighted bold. The figure also shows that there are two underlyng views involved that form the coalition workflow.

In Figure 14 the manufacturer views the assignment of the manufacturer's respective coalition workflow model entities to the corresponding workflow view model and the underlying private workflow model.


Execute Workflow - Microsoft Internet Explorer provided by SAP IT

Execute Coalition Workflow (partner: Car Manufacturer)

Diagram click behaviour: Popup status change window

Coalition Workflow (id: 2)

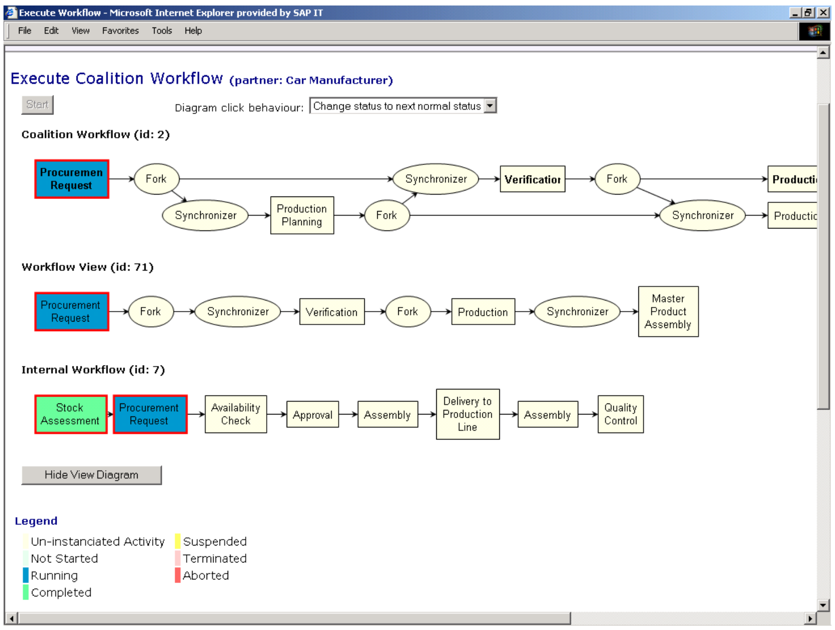
Workflow View (id: 71)

Internal Workflow (id: 7)

Legend

Un-instanciated Activity	Suspended
Not Started	Terminated
Running	Aborted
Completed	

Fig. 14. **Manufacturer**: Workflow *Model* Properties comprising Coalition, View, and Private Workflow Models


Execute Workflow - Microsoft Internet Explorer provided by SAP IT

Execute Coalition Workflow (partner: Car Manufacturer)

Diagram click behaviour: Change status to next normal status

Coalition Workflow (id: 2)

Workflow View (id: 71)

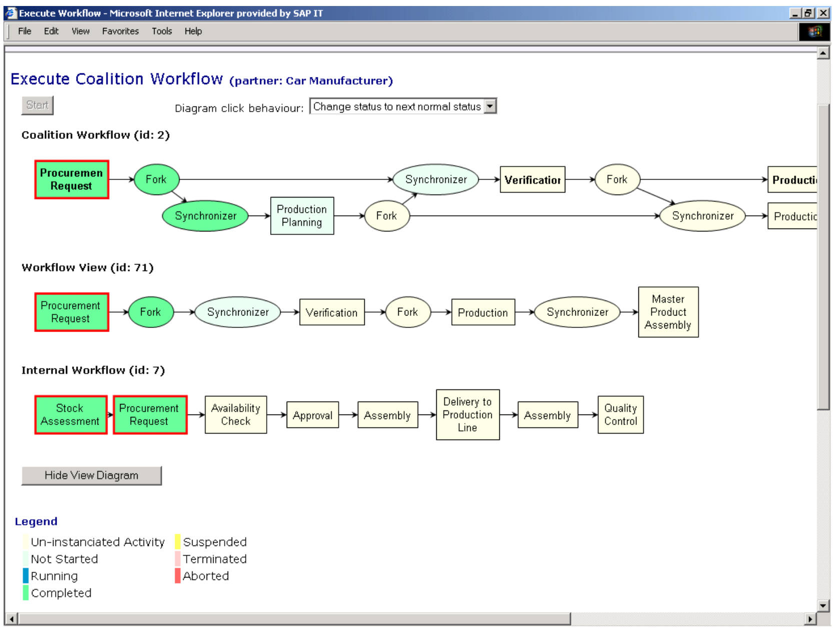
Internal Workflow (id: 7)

Legend

Un-instanciated Activity	Suspended
Not Started	Terminated
Running	Aborted
Completed	

Fig. 15. **Manufacturer**: Workflow *Instance* Properties comprising Coalition, View, and Private Workflow Instances (1 of 2)

In Figure 15 the manufacturer instantiates the workflow view instance, which, due to its inherent relationship with a private workflow instantiates a private workflow instance. In the figure a (red) border highlights the relationship of two sequential private workflow tasks (*Stock Assessment* and *Procurement Request*) to a single view task (emphProcurement Request) and the position of this view task in the coalition workflow (emphProcurement Request). Whilst the first private workflow task instance has already entered the status *completed*, the second is still *running*.


Execute Workflow - Microsoft Internet Explorer provided by SAP IT

Execute Coalition Workflow (partner: Car Manufacturer)

Diagram click behaviour: Change status to next normal status

Coalition Workflow (id: 2)

Workflow View (id: 71)

Internal Workflow (id: 7)

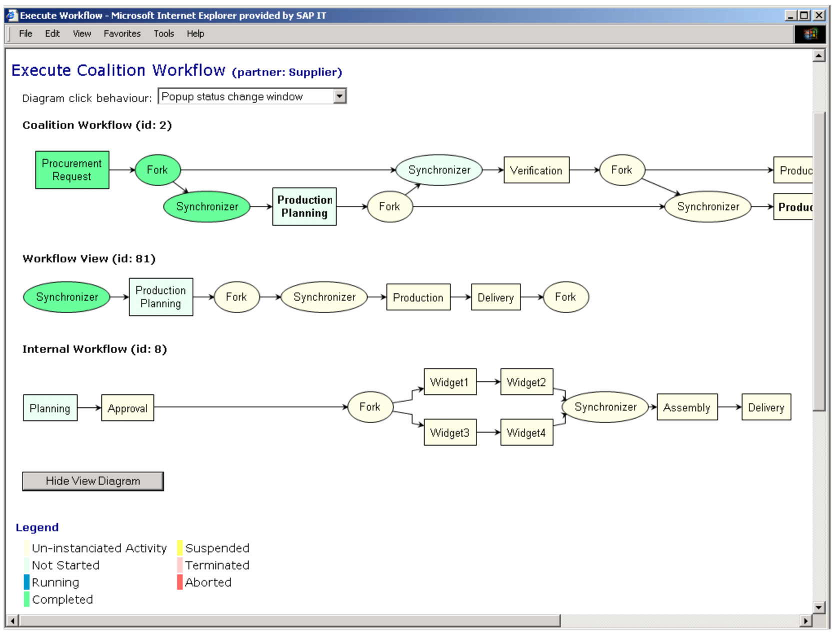
Legend

Un-instanciated Activity
Not Started
🟦 Running
🟩 Completed
🟨 Suspended
Terminated
🟥 Aborted

Fig. 16. **Manufacturer**: Workflow *Instance* Properties comprising Coalition, View, and Private Workflow Instances (2 of 2)

The adjacent workflow view task instance correctly displays the status as *running*, which is propagated to the coalition workflow instance.

In Figure 16 the progress of the manufacturer's workflow instances has reached a synchonisation point which requires the response from the supplier in order to proceed.


Execute Workflow - Microsoft Internet Explorer provided by SAP IT

Execute Coalition Workflow (partner: Supplier)

Diagram click behaviour: Popup status change window

Coalition Workflow (id: 2)

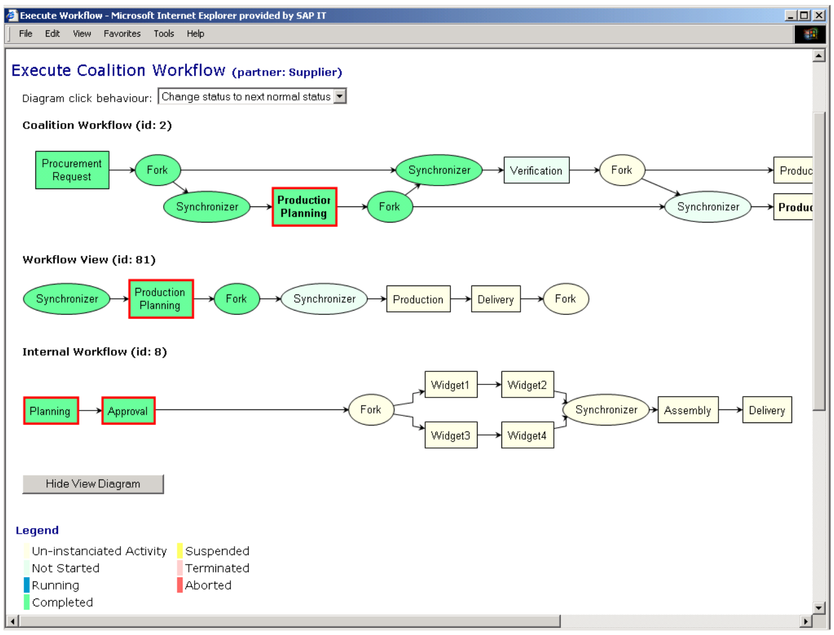
Workflow View (id: 81)

Internal Workflow (id: 8)

Legend

Un-instanciated Activity
Not Started
🟦 Running
🟩 Completed
🟨 Suspended
Terminated
🟥 Aborted

Fig. 17. **Supplier**: Workflow *Instance* Properties comprising Coalition, View, and Private Workflow Instances (1 of 2)


Execute Workflow - Microsoft Internet Explorer provided by SAP IT

Execute Coalition Workflow (partner: Supplier)

Diagram click behaviour: Change status to next normal status

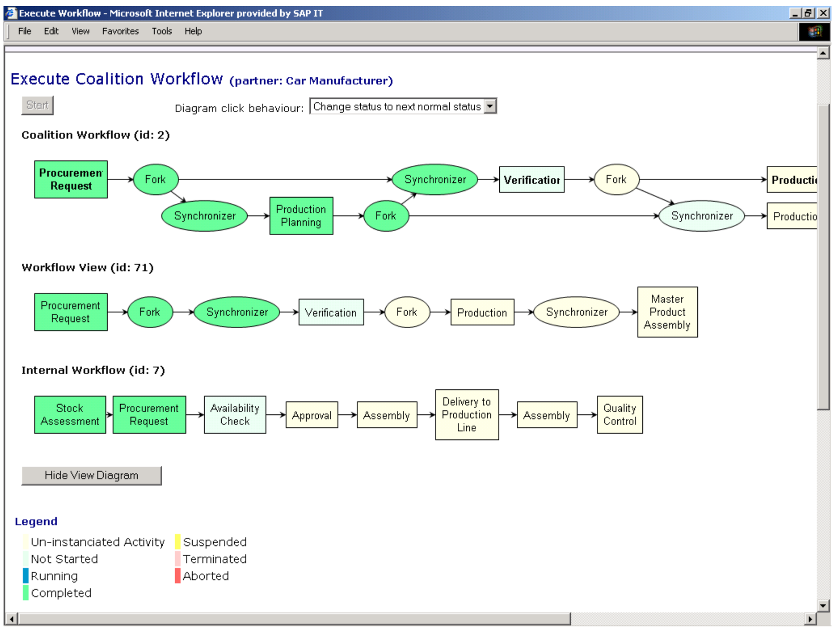
Fig. 18. **Supplier**: Workflow *Instance* Properties comprising Coalition, View, and Private Workflow Instances (2 of 2)

Figure 17 displays the monitoring information of the cross-organisational interaction from the supplier's perspective who is interacting with *another workflow engine*! Whilst the status information with regards to the coalition workflow instance is identical to what the manufacturer monitors, the supplier's underlying workflow view instance and private workflow instance are different. The supplier starts performing his task instances.

In Figure 18 the supplier himself has reached a point in which the manufacturer is notified about the results of the supplier's work. In this case, the supplier has to wait for a production confirmation message before he can continue.

In Figure 19 the manufacturer has received a message from the supplier and continues with his work.

The interaction continues further. Whilst we have demonstrated synchronous interaction so far, the scenario later requires asynchronous interaction in which the manufacturer and the supplier work in parallel to produce their respective goods (denominated *Production* in both respective workflow view instances). Our workflow engine supports synchronous and asynchronous interaction. The workflow model further provides for bidirectional communication. A view task instance can e.g. suspend its underlying private workflow task instances (e.g. as a result of an external message), whilst the private workflow instance can re-start the suspended private workflow tasks and notify the corresponding workflow view task instance about this change. This behaviour is configurable and allows reflecting actual business requirements with regards to the prioritisation of company-internal communication versus company-external communication.


Execute Workflow - Microsoft Internet Explorer provided by SAP IT

Execute Coalition Workflow (partner: Car Manufacturer)

Diagram click behaviour: Change status to next normal status

Legend

▯ Un-instanciated Activity	🟨 Suspended
▯ Not Started	▯ Terminated
🟦 Running	🟥 Aborted
🟩 Completed	

Fig. 19. **Manufacturer**: Workflow *Instance* Properties comprising Coalition, View, and Private Workflow Instances (3 of 3)

It is important to emphasize that the manufacturer and the supplier work on two workflow management systems, which operate in different organisational entities. The information exchange that is required to synchronise the workflow management systems is supported by a mediator that provides message queuing and repository services to the trading partners.

6 Conclusions and Future Work

In this paper we discussed the entities of an architecture that provides execution support for mediated and unmediated cross-organisational workflows. We argued how this architecture supports our view-based cross-organisational workflow model, which we particularly investigated with regards to state-dependencies between private workflows and workflow views. We introduced an explicit modelling approach to describe those dependencies. In this context we argued how private workflows and workflow views should be tightly coupled through state dependencies, whilst workflow views should be loosely coupled through synchronisation tasks. Finally, we demonstrated the viability of our proposed approach on the basis of an extended workflow management system that executes private workflow in equilibrium with its corresponding workflow view in the context of a coalition workflow in which it interacts with other workflow views from other partners. We proposed a Petri-net based representation as the basis for consideration of state dependencies between the tasks in a workflow and the adjacent task in a workflow view. This representation is focused on the structural aspects of a workflow, which includes the internal states of tasks and view task, and their state dependencies.

The key idea behind the introduction of workflow views that abstract from private workflows is the provision of a mechanism that allows for a multi-granular privacy of workflows. We postulate that a workflow view needs to be protected from unauthorised interaction. Outside of the context of deeper considerations in this paper are the major requirements towards any system that supports the electronic exchange of sensitive data to ensure that the data remains private, in the sense that it is protected from being observed by a third party. Whilst the workflow view approach provides for the limited visibility of certain information concerning the underlying private workflow, we have not considered topics related to the actual communication between trading partners and their workflow views. These are data integrity, which means that data is protected from being changed whilst in transit. For example, data could be modified or forged. Or a header of a valid message A could be associated with the body of a valid message B in order to create a forged message C. Messages could be replayed: A valid message could be re-used for a repeated request to IT system. Also authentication needs to be ensured: is the sender of data who he claims to be? Further, there is denial-of-service: A third party floats an IT-systems with forged IP packets such that this system becomes so busy handling these packets that it becomes very slow or unable to handle real requests.

In this paper we have mainly considered the structural aspects of a workflow and the control-flow dependencies between tasks. However, as workflows represent business processes, tasks represent actions in the real world. Actions always read, write, or alter data. Consequently, tasks have to reflect the data lifecycle, whilst workflows need to be considered with data flow in addition to control flow. We firmly believe that future work needs to address data flow in workflow modelling and execution. We briefly discuss this below.

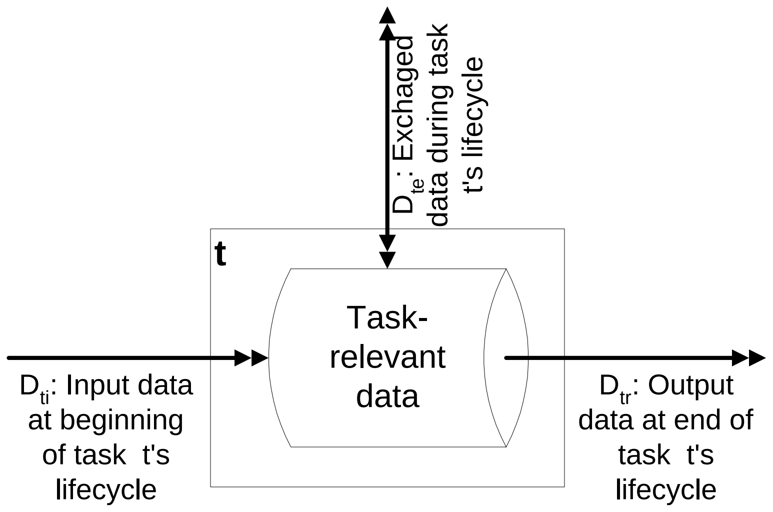

D_{te}: Exchaged data during task t's lifecycle
t
Task-relevant data
D_{ti}: Input data at beginning of task t's lifecycle
D_{tr}: Output data at end of task t's lifecycle

Fig. 20. Task-Relevant Data

In Figure 20 we consider task-relevant data. A task consumes data at the beginning of its lifecycle to enter the state *running*. During its lifecycle it exchanges data with its environment in order to achieve its business objective. At the end of its lifecycle it produces some data that is required by the next task. The union of task-relevant data from all tasks of a workflow form workflow-relevant data.

From a workflow modelling perspective we argue that the union of exchange data of a set of tasks that are represented by a single view task has to be modelled conscious of the fact that once the view-task represents many tasks that require interaction with the coalition, the coalition partner would not be aware in which order they have to exchange data with l. A better approach is to have maximum one task that requires data exchange with the coalition being represented through the view task. We note that still all tasks are able to exchange data within their own organisation, because the corresponding private workflows details are all known to its owning entity.

Finally, the issue of transactions requires further consideration. Our model is intended for transactional behaviour of private workflow tasks and their corresponding workflow view task in terms of atomicity (all state changes to a private workflow task are reflected in the corresponding view task or none of them and vice versa) and consistency (state information in private workflow task and corresponding workflow view task are consistent). However, this needs to be revisited in the context of semantic considerations of private workflows.

7 Acknowledgement

The Authors would like to thank the team members of the DKE group at the University of Queensland for their feedback and support, in particularly Shazia Sadiq. We would further like to thank Brad Clow from Workingmouse in Brisbane for his support in implementing Nehemiah. Last, but not least we would like to thank Wasim Sadiq, Sonia Lippe, and the researchers from SAP Corporate Research's Business Process Technology Research Program and our colleagues from SAP for their feedback on the concepts described in this paper.

References

[1] J. Klingemann; J. Waesch; K. Aberer. Adaptive outsourcing in cross-organizational workflows. In *Proceedings of the 11th International Conference, CAiSE 1999, Heidelberg, Germany*, pages 417–421. Springer Verlag, 1999.

[2] J. Klingemann; J. Waesch; K. Aberer. Deriving service models in cross-organizational workflows. In *Proceedings of the 9th International Workshop on Research Issues in Data Engineering - IT for Virtual Enterprises, RIDE-VE'99*, pages 100–107. IEEE Computer Society, 1999.

[3] Q. Chen. Inter-enterprise collaborative business process management. Technical report, HP Labs Palo Alto, http://www.hpl.hp.com/techreports/2000/HPL-2000-107.pdf, 2000.

[4] Dickson K. W. Chiu, Kamalakar Karlapalem, Qing Li, and Eleanna Kafeza. Workflow view based e-contracts in a cross-organizational e-services environment. *Distributed and Parallel Databases*, 12(2-3):193–216, 2002.

[5] Workflow Management Coalition. The workflow reference model. Technical Report WFMC-TC-1003, Workflow Management Coalition, 1995.

[6] Workflow Management Coalition. Workflow standard - interoperability - abstract specification, version 1.0. Technical Report WFMC-TC-1012, Workflow Management Coalition, 1996.

[7] Workflow Management Coalition. Interface 1: Process definition interchange process model specification, version 1.1. Technical Report WFMC-TC-1016, Workflow Management Coalition, 1999.

[8] Workflow Management Coalition. Workflow standard - interoperability - wf-xml binding version 1.1. Technical Report WFMC-TC-1023, Workflow Management Coalition, 2001.

[9] P. Avila; G. Putnik; M. Cunha. Brokerage function in agile virtual enterprise integration - a literature survey. In Luis M. Camarinha-Matos, editor, *Collaborative Business Ecosystems and Virtual Enterprisese, ISBN 1-4020-7020-9*, pages 65–72. Kluwer Academic Publishing, 2002.

[10] ebXML. Requirements specification version 1.0. Technical report, ebXML.org, 2000.

[11] ebXML. ebxml technical architecture specification v1.0.4. Technical report, ebXML.org, 2001.

[12] A. Lazcano et al. The wise approach to electronic commerce. *International Journal of Computer Systems Science & Engineering, special issue on Flexible Workflow Technology Driving the Networked Economy, Vol. 15, No. 5*, September 2000.

[13] Arkin et al. Web service choreography interface (wsci) 1.0. Technical report, Sun Microsystems, http://wwws.sun.com/software/xml/developers/wsci/, 2002.

[14] G. Alonso et al. Wise: Business to business e-commerce. In *Proceedings of the 9th International Workshop on Research Issues in Data Engineering - IT for Virtual Enterprises, RIDE-VE'99*, pages 132–139. IEEE Computer Society, 1999.

[15] Georgakopoulos et al. Modeling and composing service-based and reference process-based multi-enterprise processes. In *Proceedings of the 12th International Conference, CAiSE 2000, Stockholm, Sweden*. Springer Verlag, 2000.

[16] P.Muth et al. Workflow history management in virtual enterprises using a light-weight workflow management system. In *Proceedings of the 9th International Workshop on Research Issues in Data Engineering - IT for Virtual Enterprises, RIDE-VE'99*, pages 148–155. IEEE Computer Society, 1999.

[17] T. Andrews et al. Business process execution language for web services, version 1.1. Technical report, SAP, IBM; BEA; Microsoft, Siebel Systems, 2003.

[18] Object Management Group. Omg business object domain task force rfp2 submission - workflow management facility revised submission. Technical Report bom/98-06-07, Object Management Group, 1998.

[19] Object Management Group. Final report of the workflow management facility revision task force 1.2. Technical Report dtc/99-07-04, Object Management Group, 1999.

[20] Object Management Group. A uml profile for enterprise distributed object computing. Technical Report ad/2001-06-09, Object Management Group, 2001.

[21] A-W. Scheer; M. Hoffmann. From business process model to application system - developing an information system with the house of business engineering (hobe). In *Proceedings of the 11th International Conference, CAiSE 1999, Heidelberg, Germany*, pages 2–9. Springer Verlag, 1999.

[22] C. Harbilas; N. Dragios; G. Karetsos. A framework for broker assisted virtual enterprises. In Luis M. Camarinha-Matos, editor, *Collaborative Business Ecosystems and Virtual Enterprisese, ISBN 1-4020-7020-9*, pages 73–80. Kluwer Academic Publishing, 2002.

[23] K.Schulz. *Modelling and Architecting of Cross-Organisational Workflows*. PhD thesis, The School of Information Technology and Electrical Engineering, The University Of Queensland, Australia, 2002.

[24] DSTC Pty Ltd. Uml profile for enterprise distributed object computing (edoc), initial submission by dstc. Technical report, DSTC Pty Ltd, ftp://ftp.omg.org/pub/docs/ad/99-10-07.pdf, 1999.

[25] Alonso; Casati; Kuno; Machiraju. *Web Services - Concepts, Architectures and Applications*. Springer, 2004.

[26] K. Schulz; Z. Milosevic. Architecting cross-organizational b2b interactions. In *Proceedings of the 4th International Conference on Enterprise Distributed Object Computing (EDOC 2000), Makuhari, Japan. ISBN 0-7695-0867-7*, pages 92–101. IEEE Computer Society, 2000.

[27] R. Mejia; A. Molina. Virtual enterprise broker: Processes, methods and tools. In Luis M. Camarinha-Matos, editor, *Collaborative Business Ecosystems and Virtual Enterprisese, ISBN 1-4020-7020-9*, pages 81–90. Kluwer Academic Publishing, 2002.

[28] University of Twente. Crossflow contract model. Technical report, ESPRIT Crossflow EP 28653, 1999.

[29] K. Schulz; M.E. Orlowska. Architectural issues for cross-organisational b2b interactions. In *Proceedings of the International Workshop on Distributed Dynamic Multiservice Architectures (DDMA) in conjunction with the 21st International Conference on Distributed Computing Systems (ICDCS-21) Phoenix, USA. ISBN 0-7803-4503-7*. IEEE Computer Society, 2001.

[30] K. Schulz; M.E. Orlowska. Towards a cross-organisational workflow model. In Luis M. Camarinha-Matos, editor, *Collaborative Business Ecosystems and Virtual Enterprisese, ISBN 1-4020-7020-9*, pages 153–160. Kluwer Academic Publishing, 2002.

[31] K. Schulz; M.E. Orlowska. Empirical aspects of workflow interoperability. In *Witold Abramowicz, Gary Klein (eds.), Business Information Systems, Proceedings of BIS 2003, Colorado Springs, USA*, 2003.

[32] S. Angelov P. Grefen, H. Ludwig. A framework for e-services: A three-level approach towards process and data management. Technical report, University of Twente, IBM, http://wwwhome.cs.utwente.nl/ grefen/Research/Publications/techreps.html, 2003.

[33] P. Johannesson; E. Perjons. Design principles for application integration. In *Proceedings of the 12th International Conference, CAiSE 2000, Stockholm, Sweden*, pages 212–231. Springer Verlag, 2000.

[34] R. Tagg; G. Quirchmayr. Towards an interconnection model for evolution of shared workflows in a virtual enterprise. Technical report, University of South Australia, 2001.

[35] IBM Research. Crossflow architecture description. Technical report, ESPRIT Crossflow EP 28653, 1999.

[36] G. Riempp. *Wide Area Workflow Management*. Springer-Verlag London Limited, 1998.

[37] RosettaNet. http://www.rosettanet.org/, 2000.

[38] SAP. Sap business maps. Technical report, SAP AG, http://www.sap.com/solutions/businessmaps/c-businessmaps/, 2001.

[39] M.-T. Schmidt. The evolution of workflow standards. *IEEE Concurrency Journal*, pages 44–52, July 1999.

[40] G. Preuner; M. Schrefl. Behaviour-conistent composition of business processes from internal and external services. 2002.

[41] K. Schulz. Monitoring of cross-system workflows (in german), 1996.

[42] K. Schulz. Implementation of the distributed workflow service, d303. Technical report, ESPRIT Vega EP 20408, http://cic.sop.cstb.fr/ILC/ecprojec/vega/workpack.htm, 1998.

[43] D.Liu; M. Shen. Modeling workflows with a process-view approach. In *Proceedings of the 7th International Conference on Database Systems for Advanced Applications*, pages 260–267. IEEE Computer Society, 2001.

[44] S. Aissi; P. Malu; K. Srinivasan. E-business process modeling: The next big step. *IEEE Computer Journal*, 35(5):55–62, May 2002.

[45] R. Steinmann. Demonstration of virtual lse enterprise - vega 5th review demonstration. Technical report, ESPRIT Vega EP 20408, 1999.

[46] R. Steinmann. Public final report. Technical Report PFR-01, ESPRIT Vega EP 20408, http://cic.sop.cstb.fr/ILC/ecprojec/, 1999.

[47] uddi.org. http://www.uddi.org/, 2003.

[48] Umar. A framework for analyzing virtual enterprise infrastructure. In *Proceedings of the 9th International Workshop on Research Issues in Data Engineering - IT for Virtual Enterprises, RIDE-VE'99*, pages 4–11. IEEE Computer Society, 1999.

[49] W-J. van den Heuvel; H. Weigand. Cross-organizational workflow integration using contracts. In *Proceedings of the Business Object Component Workshop of the ACM Conference on Object-Oriented Programming, Systems, Languages, and Applications*. Springer Verlag, 2000.

[50] W3C. Web services description language (wsdl) 1.1. Technical report, W3C, http://www.w3.org/TR/wsdl, 2001.

[51] J. Becker; M. zur Muehlen. Towards a classification framework for application granularity in workflow management systems. In *Proceedings of the 11th International Conference, CAiSE 1999, Heidelberg, Germany*, pages 411–416. Springer Verlag, 1999.