View-based Interorganizational Workflows

Johann Eder, Nico Kerschbaumer, Julius Köpke, Horst Pichler, Amirreza Tahamtan

Abstract: *We propose a new architecture for modeling and enacting of interorganizational business processes which captures both control flow and data flow aspects of web service orchestrations and choreographies. A major modeling concepts is a view mechanism to balance the problems of openness, privacy and coupling in cross-organization interactions. The view concept cares both for process abstraction and for data transformation between business partners. To cope with interoperability problems that arise from different data formats and internal schemas used by different choreography participants we support transformation of XML-based data types through our views. The approach allows a to enact interorganizational business processes in a fully distributed way without the need for any central coordinator.*

Key words:

INTRODUCTION

Interorganizational workflows support the cooperation and communication of different autonomous entities (companies, organizations). A lot of research has been conducted in the recent years to develop approaches to support process communication with web technologies [1, 2, 3] in particular recently with SOA-based protocols between business partners [4] - business processes as service compositions. There are two main approaches: (a) orchestrations where a local (private) process is executed and controlled by a single orchestration engine and communicates with external services, that can be entry or exit points of partner processes; or (b) choreographies which define a communication protocol -- a sequence of message exchanges -- between collaborating business partners, which can be interpreted as a decentralized global process [5] (cmp. also to executable and abstract BPEL definitions [6, 7]).

Still, cooperation is a very sensitive matter. Partners have to work together to fulfil a common business goal that is profitable for both, but they do not want to grant a public view on their process internals and usually it is not necessary to reveal theses internals to model a global process. E.g., the buyer does not need to know how the seller interacts with the shipper, as this would unnecessarily clutter the global process for all participants. Furthermore, certain message exchanges shall remain private between two partners, e.g. the protocol between the bank-institute and a buyer. This calls for a modeling architecture which allows to model in great detail which information is disclosed to which partner, and still is able to support the modeling of complex interorganizational workflows.

A very promising solution that allows a good balance between openness and privacy is the concept of workflow views. A workflow view is created by applying structural operations onto the original workflow, like hiding activities or aggregating a set of activities to a new step. We want to apply views for collaborations between business partners with the goal to incorporate private orchestrations into a global choreography (or vice versa), as well as for collaborations where views are applied to keep privacy while still allowing enough openness to communicate with external partners. Partners that need to cooperate provide views on their private internal processes or build their internal processes such that they can be connected to existing views and then solely communicate through their partners provided views [8].

CompSysTech'11, June 16–17, 2011, Vienna, Austria.
Copyright ©2011 ACM 978-1-4503-0917-2/11/06...$10.00.

Research conducted in recent years primarily aimed at control flow aspects, like providing operations for the generation of views and checking the structural validity, which are by now well understood and handled. We identified several additional essential issues that must be considered during the design as well as during the run time of a view based collaboration approach.

A missing piece in the modeling phase is the consideration of *data aspects*, like the specification of data structures (schemas) and message *transformations* between different schemas. As data flow problems may result in non-functional processes, even if the control flow has been modeled correctly (see also [9]), it is required to *validate the data flow* between internal and external processes. Furthermore, a modeling language should support top down (from global process via views to private processes), bottom up (vice versa), or hybrid *design approaches*. This also includes features like semi-automatic generation of views and processes as well as their structural validation. And to our knowledge currently no modeling technique exists that supports *reusability*, like building views on top of other views or reusing existing views in new choreographies (see also [10]) and to uniformly describe these kind of scenarios.

Additionally several issues have to be considered at run time -- within the engines - as they influence the way local processes interact in the global process via views and which information views must have access to. During execution one must provide the means to *support structural and data mapping* between executed internal orchestrations and views and vice versa. Furthermore, *correlation issues* arise, because the view needs to be able to deliver data received from an external provider to the right step in the correct process instance or send a message to the right partner interface.

In existing approaches, if a new partner wants to join the choreography or a partner wants to change the way he communicates, a new model must be distributed to and implemented by all partners. This is not always required, because it will not always concern all partners. Therefore we claim that *extensible and dynamic coupling* (late binding) must be supported in such an architecture. Late binding of partner interfaces, which is also required for dynamic service selection, is not supported in existing concepts.

Finally, decomposing the global process into several interdependent parts results in the loss of a view on the current state of the global process as it degrades to a sequence of service calls between partners. Therefore we need tools to *monitor the progress within the global process*. The overall goal of our view-based approach is to tackle all issues identified and described above. This paper lays the basis by introducing a corresponding workflow definition language and a view generation language along with a description of an architecture containing all components required to reach this ambitious goal. An earlier version of this paper was presented in [19].

MODELLING INTERORGANIZATIONAL BUSINESS PROCESSES

In this chapter we present a common multi-partner collaboration scenario and show how our view-based approach can be used to handle the communication and data transformations through views.

We introduce the notation of our elements used in the following examples in figure 1. We use the concept of activities that can be reused in activity steps, XML documents to store data, swim lanes to model a partner's private orchestration and views, plain line connectors for the internal control flow and dashed lines for the external control flow. An occurrence of a XML document in a plain or dashed line means that this document is transferred to the corresponding activity step.

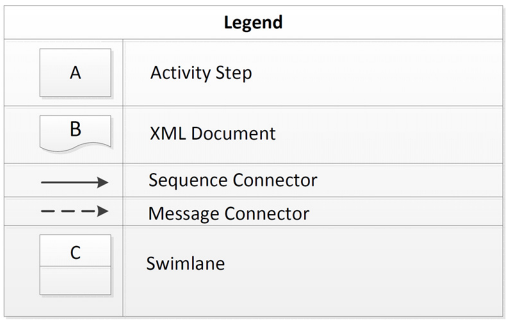


Fig. 1. Notion of elements

In figure 2 we present an example of three interacting partners to illustrate how our approach works. In our scenario a *buyer*, a *seller* and a *shipper* interact with each other to fulfil a common business goal. The basic process is as follows, a seller offers products and ships them to buying customers via a shipper. The buyer also interacts with the shipper to either check the status of a shipment. The buyer offers two views, one for the seller(*BView1*) and one for the shipper(*BView2*); the seller offers one view for the buyer(*SView1*) and one for the shipper(*SView2*); and finally the shipper offers three views, one for the seller(*SHView1*), one to allow buyers to check the delivery status of a shipment(*SHView2*) and one for the transport interaction with a buyer(*SHView3*). Also notice that the view(*SView1*) of the seller has its exposed steps renamed to German, this shows the possible need for a rename functionality.

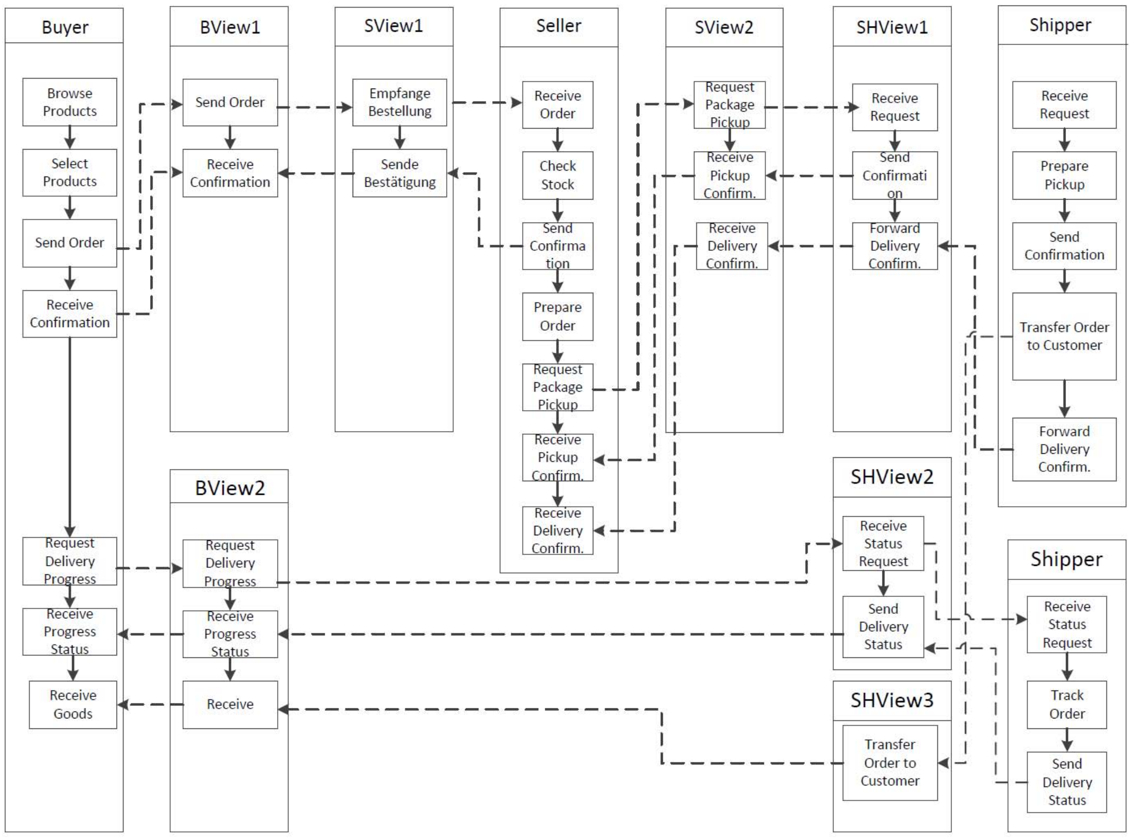


Fig. 2. Example showing a view based communication

Figure 3 shows how an *OrderBuyer* document is send to the seller and transformed in the view to an order document type that the seller can process. After the seller received the document and finished his internal steps he sends an confirmation to the buyer which is again an XML document that gets transformed in the sellers view to a document type the buyer expects.

An example how the document instances are exchanged and transformed in the buyer/seller example of figure 3 is shown in figure 4. The buyer merges the forename and surname elements and transforms the date into a different format, while the seller uses internally a confirmation document and the order documents which are merged into a single document because the receiving step of the buyer only expects a one document.

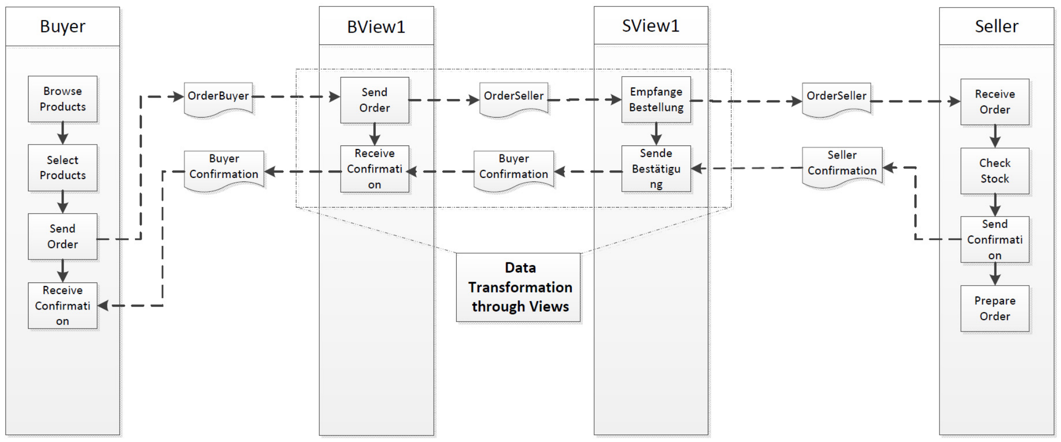
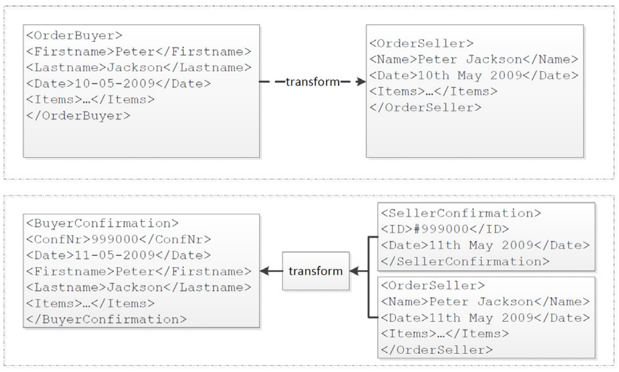


Fig. 3. Example communication with data transformation



```
<OrderBuyer>
<Firstname>Peter</Firstname>
<Lastname>Jackson</Lastname>
<Date>10-05-2009</Date>
<Items>…</Items>
</OrderBuyer>
```

```
<OrderSeller>
<Name>Peter Jackson</Name>
<Date>10th May 2009</Date>
<Items>…</Items>
</OrderSeller>
```

```
<BuyerConfirmation>
<ConfNr>999000</ConfNr>
<Date>11-05-2009</Date>
<Firstname>Peter</Firstname>
<Lastname>Jackson</Lastname>
<Items>…</Items>
</BuyerConfirmation>
```

```
<SellerConfirmation>
<ID>#999000</ID>
<Date>11th May 2009</Date>
</SellerConfirmation>
```

```
<OrderSeller>
<Name>Peter Jackson</Name>
<Date>11th May 2009</Date>
<Items>…</Items>
</OrderSeller>
```

Fig. 4. XML document transformation

FEDERATED CHOREOGRAPHIES

The typical scenario for an interorganizational business process is a shared choreography and a set of private orchestrations. However, this scenario misses an important fact and is not fully suited to cope with all real life applications. Imagine a simple online shopping scenario. When shopping online a buyer takes part in a choreography with a seller, a credit card provider and a shipping company. The steps that the buyer needs to know of the process are that he orders products at the seller, pays via a credit card company and receives the order from a shipping company. At the same time the seller and shipper take part in several other choreographies which are not visible to the buyer, like, e.g., the seller and credit card company have agreed on a process for handling payment through a bank. As one can see from this example, more than a choreography may be needed to reach the goals of a global business goals. The choreographies will overlap in some parts and it is not always possible to combine them into a single global choreography. The choreographies also have to be realized by orchestration provided by participating partners. Even if the combination of all choreographies into one choreography would be possible, the separation offers obvious advantages. A nouvelle approach for interorganizational workflow modeling called *federated choreographies* was proposed in [10, 8]. A federated choreography consists basically of two layers:

- Bottom Layer: Contains the orchestrations that realize the choreographies in the upper layer. An orchestration provides several views for different interactions with other partners. The partner interactions are reflected in the choreographies.

- Upper Layer: Consists of the federated choreographies shared between business partners. A choreography is formed through the views of orchestrations, which means that only activities contained in the view are part of the choreography. A supporting choreography describes parts of a supported one in more detail. The global choreography supports no other choreography and captures the core of a business process while the choreographies that support the global choreography are capable the provide enough details for implementations.

The federated-choreography approach is fully distributed and there is no need for a centralized coordination. Each partner has local models of all choreographies in which it participates. All local models of the same choreography are identical. By having the identical local models of choreographies, partners know to which activities they have access, which activities they have to execute and in which order. In addition, partners are aware when to expect messages and in which interval they can send messages. In other words, the knowledge about execution of the model is distributed among involved partners and each partner is aware of its duties in the course of process execution. Hence, there is no need for a super-user or a central role that possesses the whole knowledge about execution of the process. Rather this knowledge is distributed among participants and each partner knows what he needs to know. Additionally, each partner holds and runs its own model of the orchestration. Federated choreographies are more flexible than typical compositional approaches used in proposals like WS-CDL[11] and it closes the gap between choreographies and orchestrations by providing a coherent and integrated view on both choreographies and orchestrations. Federated choreographies offer obvious advantages such as protection of business know-how, avoidance of unnecessary information in the global process, extendability and lastly they offer a coherent and uniform modeling for both choreographies and orchestrations.

PROCESS AND VIEW DEFINITION

Our approach for view-based federated choreographies is grounded on an integrated workflow model that allows a consistent modeling of workflows and views. For the creation of workflows we use our language, which is called distributed workflow definition language (DWDL). For the generation of views a workflow view definition language (WfVDL) is used which can be applied on workflow definitions. Due to space limitations we cannot include the full specification of the language and would like to refer to reader to [12]. A workflow view is defined to be a view of an internal workflow and is created for an external partner. The consistent modeling of workflows and views allows the definition of different relations between workflows and views. In this paper we will present the different relations but we will not provide the formal definitions due to space limitations.

Figure 5 which shows an example how a global business process can be modeled with our approach and explain the relationships between the different types of workflows, i.e. choreographies, orchestrations and views. The process consists of a choreography WF_1, three orchestrations WF_A, WF_B, WF_C and a set of views among them VX_i. We introduce the following relations between workflows:

- The **isViewOf** relation denotes that a workflow is a correct view of another workflow, e.g. WF_A *isViewOf* WF_B. Since we treat choreographies, orchestrations and views as workflows the relation can exist between a choreography and a view on the choreography, between an orchestration and a view on the orchestration or a view can also be a view of another view.
- The isEqual relation denotes that two given workflows are equal, meaning that their

set of nodes has the same labels, matching parameter types and the workflows posses an equal structure.

- The isCounterPart relation denotes that two workflows can form a correct communication protocol, meaning that for every sending step in a workflow WFA there exists a corresponding receiving step in a workflow WFB.

The choreography WF_1 in figure 5 is split into three views, whereas each view is defined for a certain role of a partner. A partner that wants to participate in the choreography has to be able to provide a view based on his orchestration that has to be equal to the view published by the choreography, e.g., VA_O *isEqual* VA_C. A partner that is part of the choreography might have to interact with more than one partner to realize it's part of the choreography, e.g., the orchestration of partner *B*, WF_B can provide a view that is equal to the choreography view VB_O, which is needed to take part in the global process and furthermore it provides two view VB_1, VB_2 for the communication with partner *A* and partner *C*. The views VB_1 and VB_2 have to be view of the view VB_O, because the choreography defines the communication among partners. Between two interacting partner views the counterpart relation must hold, e.g.,VA_1 *isCounterPart* VA_2, which means that the protocol between partner *A* and partner *B* is valid.

In our model there are two main approaches, namely top-down and bottom-up, to build workflows:

- When the top-down approach is used at first a global choreography is defined in form of a workflow definition. Afterwards views for all involved external roles are created upon the global choreography. The next task is the creation of the private orchestrations of the involved parties (if they do not yet exist). In the last step views are created upon the orchestrations. In this case the views of the orchestrations must be equal to the corresponding views of the global choreography. At runtime these views are used in order to establish the communication between the partners.

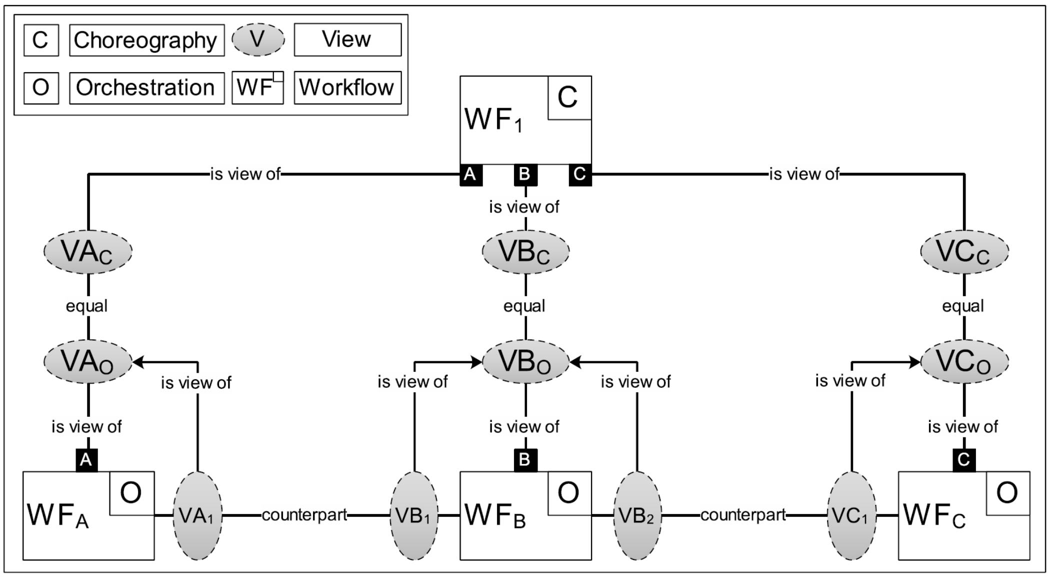


Fig. 5. Workflows, Views and their Relations

- Another approach is the bottom-up approach where different partners already have their workflows and want to cooperate in a peer to peer fashion. In this case views for each involved external role need to be created and integrated [13]. The views must be counterparts of each other. Because each view allows the definition of transformations it is up to the involved parties (maybe market strength) who actually performs the transformations. It is even possible to use a common data-exchange format between the partners that is not used by any partner internally. In this case the transformation are defined in both views. This is most beneficial if a large number of potential partners exists.

With the concept of federated choreographies we can extend our modelling capabilities even further. Since we treat all processes as workflows, a choreography can also be a view on another larger choreography where it only realizes a small portion, e.g. WF_1 of figure 5 can again be a view of a global choreography. On the orchestration level the partner executing WF_A can provide additional views that take place in different choreographies, which again can be chained with other choreographies or orchestrations via views. This view chaining can be arbitrarily nested and combined to model very complex global processes without participants cluttering each others process views with unnecessary information.

SYSTEM ARCHITECTURE

Our architecture consists of build-time and run-time components as visualized in Figure 6.

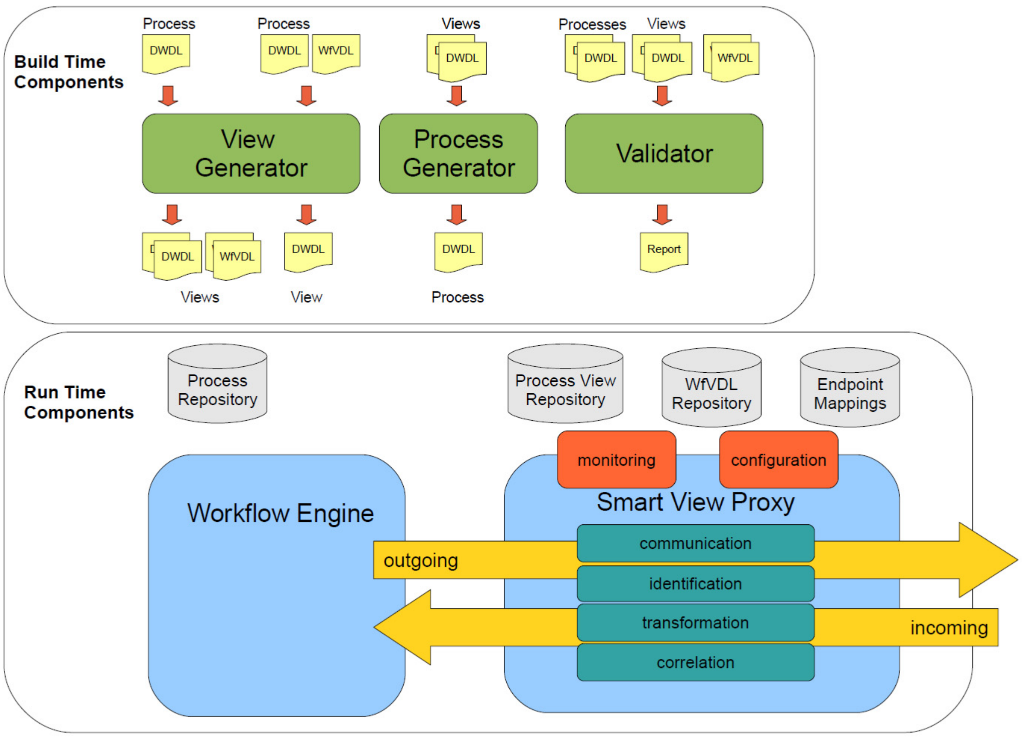


Fig. 6: System Architecture

The build time part consists of diverse generators and validation components which support workflow designers when applying a top down, bottom up, or mixed design approach. The *View Generator* supports two types of view generation. The first one takes a single process definition (e.g., a global process) as input and generates multiple view-stubs from it by exploiting the role information of communication steps. Therefore it automatically generates one view per role in the original process definition. Workflow designers may later add details about exposition and aggregation, as well as data transformation. The second type can be applied to generate a view from a process definition and a corresponding WfVDL-document. The *Process Generator* is used to generate a process definition from multiple process views. *Validators* are used to support the designers by validating the structural integrity and conformance of message(-types) of a given set of interdependent process and view definitions.

Generated or manually created DWDL documents, WfVDL documents are stored in diverse Repositories for further usage at run time. Additionally it is required to store information about endpoints (which services to invoke for communication steps of the process) in the Endpoint Repository. These may later be exchanged without changing the process or view definitions.

Required run time components are a *Workflow Engine*, which drives all private processes according to the workflow's DWDL document, and a *Smart View Proxy*, which handles message exchanges and transformations. The basic idea of this architecture is to free the workflow engine from communication and transformation issues which may differ from partner to partner -- therefore relieve it from the need to implement a new process each time a partner changes. Therefore the main task of the Proxy will be the *identification* of partners for incoming and outgoing messages, selecting and performing the appropriate *transformation* procedures and the *correlation* between process and step instances with the corresponding view and performing the actual *communication* (calling services, receiving messages). Additionally it is extended by two run time modules: one to *monitor* processes on a global scope - of course the level of detail provided by this component is dependent on the knowledge provided in views used by a process; and a second module to *configure* the information of collaborating partner processes or the accessed communication interfaces respectively of already deployed processes.

RELATED WORK

The authors of [14] introduce a two-step approach for the construction of customized process views for cross-organizational collaborations. The first step creates a non-customized process view based on an internal process by aggregating internal activities which the provider wishes to hide. In the second step a customized process view is constructed by aggregating and omitting activities from the non-customized view. The customized view is created by a consuming partner based on the non-customized view of the providing partner, thus allowing to filter out unwanted parts of the process.

[3] presents a three step approach to inter-organizational workflow cooperation. In the first step each organization identifies partners with knowledge that can be gathered to to carry out tasks which are not within the range of one organization. In the second step partners negotiate their roles in the future virtual organization and the communication protocol of their workflows. In the third and last steps were the actual workflow cooperation happens, the authors follow an approach of a trusted third party which acts as a certification authority that provides validation of participants.

[15] introduced a method to verify interorganizational workflows represented as workflow nets. A workflow is correct if all participates private process and the global workflow satisfy the correctness criteria introduced in [16]. The notion of soundness is used as the correctness criteria for internal workflows and the IO-soundness property for interorganizational workflows. An unfolding function connects all local workflows of a participant with a start and an end transition.

[17] presents a formal notion for a multiparty contract approach for agreeing and implementing interorganizational processes. The contract describes the overall business process and the duties of all involved participants. The authors use a process-oriented contract which can be seen as the composition of the public views from all partners. Based on the resulting contract all participants implement their, i.e. their private view on the global process, in such a way that it agrees on the contract. Furthermore the notion of accordance between a private view and a public view is given which if satisfied guarantees that the process is deadlock-free and that it will always terminate properly.

[18] use workflow views for realizing the communication and cooperation of autonomous processes in an interorganizational setting. The authors differ between a shared global process, called the coalition workflow and workflow views on their internal private workflows. A view only contains the necessary parts of the private workflow to

allow the communication with external partners in the coalition workflow. The architecture in this approach has a tight coupling between the private workflows and its derived views and a loose coupling between partner views in the global coalition workflow.

The following aspects distinguish our approach from the work discussed: We want to fully support and handle data flow issues, support the monitoring of the global process by its participants and we will go a step beyond ordinary choreographies and support the concept of federated choreographies as described in [8, 10]. For the view generation we introduced a language utilizing an expose statement in contrast to the hide operation of the approaches discussed above. The expose operation is more powerful since it allows one to define mappings and data transformation between otherwise incompatible send and receive activities of different business partner of a choreography. Beside build time aspects which validate workflows and views on workflow we are also considering runtime aspects that concern dynamic service rerouting to allow on-the-fly transformation and correlation issues. Since our views can include not only the parts that are necessary for the communication between partners, but also internal process steps for information purposes, it is possible for external participating partners to monitor how the progress in the global process is advancing.

CONCLUSIONS AND FUTURE WORK

We propose an architecture for modelling and enacting interorganizational business processes where choreographies, orchestrations and views are modelled as workflows – the views provide the interface between local processes and interorganizational processes.

Instead of relying on ordinary service choreographies, we base our system on the concept of view-based federated choreographies, which aims at process design from diverse starting points (top down, bottom up, mixed). Views can be generated from process descriptions and vice versa, the structural integrity and data integrity of processes and corresponding views can be validated, and the validity of resulting interaction protocols can be checked.

The structure of the presented system architecture aims at a clear separation of process execution and communication semantics. The Smart View Proxy is responsible for the communication between the internal process and external partners, specifically for the *identification* of partners for incoming and outgoing messages, selecting and performing the appropriate *transformation* procedures and the *correlation* between process and step instances with the corresponding view. This allows us to (a) provide data to partners in a form they can process, (b) alter received data in such a way that the internal process can use them, and (c) multiple partners can agree on a standard communication schema for which all participants have to provide transformations from their internal representation to the standard and vice versa. This separation enables us to change the binding to partners and their service interfaces, even during run time, on the fly, without the need to renegotiate and build the whole process anew.

The Smart View Proxy is a stateful component of the architecture which communicates with other engines, partner interfaces, or other Smart View Proxies. It will also contain a component which allows to drill into the structure of views and corresponding processes as well as to monitor the current state of pending communication sequences as parts of local or global processes.

REFERENCES

[1] Groiss, H., Eder, J.: Workflow systems for inter-organizational business processes. ACM SIGGroup Bulletin 18 (1997)

[2] Aalst, W.M.P.v.d.: Process-oriented architectures for electronic commerce and interorganizational workflow. Information Systems 24(8) (1999)

[3] Chebbi, I., Dustdar, S., Tata, S.: The view-based approach to dynamic interorganizational workflow cooperation. Data Knowl. Eng. 56(2) (2006)

[4] Alonso, G., Casati, F., Kuno, H., Machiraju, V.: Web Services: Concepts, Architectures and Applications. Springer-Verlag, Berlin, Germany (2003)

[5] Peltz, C.: Web services orchestration and choreography. IEEE Computer 36(10) (2003)

[6] OASIS: Web services business process execution language (wsbpel)

[7] Louridas, P.: Orchestrating web services with bpel. IEEE Software 25 (2008)

[8] Tahamtan, A.N., ed.: Web Service Composition Based Interorganizational Workflows: Modeling and Verification. Südwestdeutscher Verlag fuer Hochschulschriften AG, Saarbrücken, Germany (2009)

[9] Combi, C., Gambini, M.: Flaws in the flow: The weakness of unstructured business process modeling languages dealing with data. In: Proc. CoopIS(2009)

[10] Eder, J., Tahamtan, A.N., Lehmann, M.: Choreographies as federations of choreographies and orchestrations. 25th Int.Conf. on Conceptual Modeling (ER), (2006)

[11] Kavantzas, N., Burdett, D., Ritzinger, G., Fletcher, T., Lafon, Y., Barreto, C.: Web services choreography description language version 1.0 http://www.w3.org/ TR/2005/CR-ws-cdl-10-20051109/.

[12] Kerschbaumer, N.: Distributed workflow and view definition languages. TR-ISYS (2010)

[13] Frank, H., Eder, J.: Integration of statecharts. In: Proc. CoopIS, IEEE Computer Society (1998)

[14] Eshuis, R., Grefen, P.W.P.J.: Constructing customized process views. Data Knowl. Eng. 64(2) (2008)

[15] Aalst, W.M.P.v.d.: Interorganizational workflows: An approach based on message sequence charts and petri nets (1999)

[16] van Aalst, W.M.P.v.d.: Modeling and analyzing interorganizational workflows. In: CSD '98: Proceedings of the 1998 International Conference on Application of Concurrency to System Design, Washington, DC, USA, IEEE Computer Society (1998)

[17] Aalst, W.M.P.v.d., Lohmann, N., Massuthe, P., Stahl, C., Wolf, K.: Multiparty contracts: Agreeing and implementing interorganizational processes. Comput. J. 53(1) (2010)

[18] Schulz, K.A., Orlowska, M.E.: Facilitating cross-organisational workflows with a workflow view approach. Data & Knowledge Engineering 51(1) (2004)

[19] Eder J., Kerschbaumer N., Köpke J., Pichler H., Tahamtan N.: Federated Choreographies for Interorganizational Workflows. II Krajowa Konferencja Naukowa Technologie Przetwarzania Danych, Poznan, 21-23 czerwca (2010).

[20] Eder, J., Gruber, W.: A meta model for structured workflows supporting workflow transformations. In: Advances in Databases and Information Systems, Springer (2002)

ABOUT THE AUTHORS

Johann Eder, Nico Kerschbaumer, and Julius Köpke are with the Department of Informatics Systems, Alps-Adria Universität Klagenfurt, Universitätsstr. 65-67, 9020 Klagenfurt. Email: firstname.lastname@aau.at

Horst Pichler is with Ilogs GesmbH, 9020 Klagenfurt. Email: horst_pichler@gmx.net

Amirreza “Nick” Tahamtan is with the Department of Software Technology & Interactive Systems, Vienna University of Technology, Austria. Email: tahamtan@ifs.tuwien.ac.at