Exchanging Process Specifications for Identifying Cooperative Information Systems

Carlo Simon[(1)], Andreas Winter[(2)]
(1) Institute for Management / (2) Institute for Software Technology
University Koblenz-Landau, Germany
{simon | winter}@uni-koblenz.de

I. Abstract

In cooperative information systems, private and public elements of business processes are separated from each other. Hereby, public elements are published for finding and working with cooperation partners; private elements are covering strategic knowledge. This architecture requires several types of integration which must be enabled by an appropriate exchange format for the underlying graph representation. The paper demonstrates at the example of GXL the definition of such a language.

II. Introduction

One central property of cooperative information systems is the exchange of information objects. Such an exchange, however, must base on a protocol that has to be negotiated among the business partners. Consequently, a cooperation cannot start before the partners have implemented business processes that satisfy such a protocol.

Our approach within this paper bases on the idea of publishing such parts of the business processes which (1) are required for a cooperation but (2) do not disclose strategic secrets. We then use these public elements for identifying business partners with comparable process structures, simplify the protocol finding and integrate their processes.

There exist several approaches to the description of business processes, especially Petri net (cf. [17], [4]) based approaches like Workflow nets introduced by van der Aalst (cf. [2]), or Event-driven Process Chains introduced by Scheer (cf. [18]). While the latter are more a visualization than a formal and executable model [11], Workflow nets can be used for both: high-level descriptions of business processes and executable specifications for Workflow Management Systems. For this, Workflow nets must be sound (i.e. unambiguously interpretable, cf. [1]), however, Dehnert has demonstrated that relaxed sound Workflow nets can be transformed into a sound and consequently executable form [8]. So, soundness is not a primary modeling goal.

We consider the exchange of business process models formalized as Workflow nets and information objects formalized as UML Class Diagrams. Since our models are represented as graphs, beside a specification and a meta model for the different types of graphs used, we also need an exchange language. We decided on GXL [26] as an appropriate language, keeping in mind that there also exist other languages like EPML (Event-driven Process chains Markup Language) [14] or PNML (Petri Net Markup Language) [5] for the exchange of business process models. Whereas EPML and PNML are oriented towards exchanging (only) Event-driven process chains or Petri-nets, respectively, XMI (XML Meta data Interchange) [16] provides a general means for exchanging visual models.

Like XMI, GXL also offers a metamodel based adaptive interchange format. In contrast to XMI, GXL is strongly graph-based and designed simpler. This leads to much sparser documents and GXL requires only one domain independent document type definition. In contrast to EPML and PNML, GXL is not restricted to one control-flow oriented notation. Using and combining suitable GXL metaschemas provides an integrated support for exchanging various views on business process models, like control view, data view, information view, and structural view. However, these criteria do not rule out in principle the possible use of other exchange languages.

Moreover, we need a formalism which allows to prove the integratebility of business processes among each other. Here, we use the join operator of the Logic of Actions (LoA) to answer this question (cf. [19], [13]). We motivate our approach by an example.

Our paper is structured as follows: in the next section, we use negotiations as an example for cooperative information systems to motivate for our process oriented view on such systems. Hereby, we use Workflow nets to model the underlying process structure and Predicate/Transition nets whenever in such processes the access onto business objects is required. In the following section, we use a metamodel to define a GXL exchange format for our

models and demonstrate the resulting file structure with the aid of an example. Afterwards, the integration steps which have become possible due to the existence of such an exchange format are formulated in a Logic of Actions. The paper ends with some concluding remarks.

III. Modeling Negotiations as Exemplary Business Processes

Models of business processes in general comprise the following elements: a model of the *organizational structure*, a model of the *information objects* used, and *data* and/or *control flow* models of the actual processes. Hereby, the tasks described within the process models are used to link the distinct models together by allocating the required resources for this task - the human resources are allocated on the base of the organizational model, others on the base of the information objects.

The existence of several exchange formats for business objects from EDI (cf. [22]) to XMI (cf. [16]) demonstrates that these objects are in the center of the investigations on cooperative information systems. At the example of electronic negotiations, we demonstrate the importance of the actual processes for cooperations (although the negotiations themselves may not be cooperative). Hereby, we model the negotiation processes of the participating parties with the aid of Workflow nets.

Definition 1 (WF-net, WF-system) A Petri net $\mathcal{N} = (P, T, F)$ consisting of places P, transitions T and a flow relation F between them is a WF-net, if:
(i) $\mathcal{N}$ has two special places, i and o. Place i is the only source (${}^\bullet i = \emptyset$) and place o is the only sink ($o^\bullet = \emptyset$).
(ii) Let $t^* \notin T$. The short-circuited net $\overline{\mathcal{N}} = (P, T \cup \{t^*\}, F \cup \{(o, t^*), (t^*, i)\})$ is strongly connected.

A WF-system $(\mathcal{N}, i)$ is a WF-net in which place i is initially marked.

We assume that the negotiation partners model their negotiation processes independently from each other. A requester Q is giving an initial offer, expecting a reaction of a responder P to this offer (acceptance ($accept_P$), termination ($terminate_P$), or modification ($change_P$)). In the case of acceptance or termination, the negotiation ends at this stage. Otherwise, the requester evaluates the current binding offer ($evaluate_Q$) followed by accepting the modified offer ($accept_Q$), by terminating the entire negotiation ($terminate_Q$), or modifying the offer himself/herself ($modify_Q$ and $change_Q$). Now, the described process can repeat.

A differentiation into a *modify* and *change* transition is due to a separation of the process of finding a counter offer from the actual process of change: the first is typically private, the second public.

The responder might behave correspondingly except (1) s/he initially receives an offer and (2) due to political reasons wants to be the one who accepts an offer, i.e. s/he wants to have the final word in the case of acceptance. Figure 1 visualizes the (independent) processes of requester and responder. As an abbreviation, we leave away output places which must be located in the postset of such transitions which are currently without a postset.

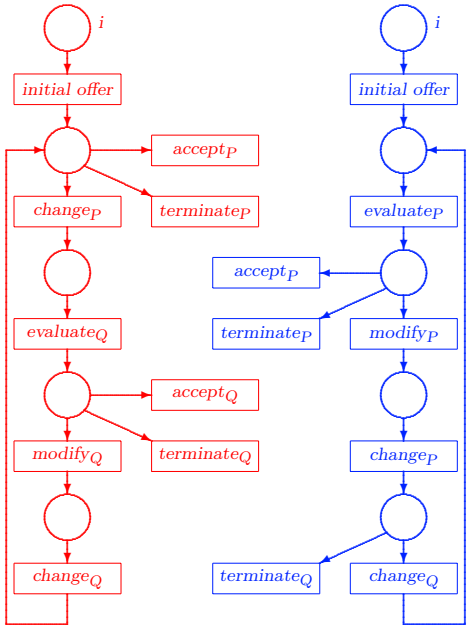

Fig. 1. Processes of requester (left) and responder (right)

The information objects exchanged throughout a negotiation is the current binding offer. In accordance with the general definition of Simon and Rebstock [21] the structure of offers in multi-attributed negotiations consists - beside general offer attributes - of the items to be traded and the items to be traded for, one of these is typically money. Therefore, each item of a binding offer has a *Direction* attribute specifying whether the item is issued from the requester to the responder ($Q \rightarrow P$) or vice versa ($P \rightarrow Q$). Figure 2 shows the corresponding data model.

The negotiation partners assess the attractiveness of the current binding offer individually based on their private needs and preferences. The access onto these information is implemented within the tasks *evaluate* and *modify*. Although the knowledge regarding the occurrence of these tasks is public, the actual decision and especially the criterions for these decisions are not.

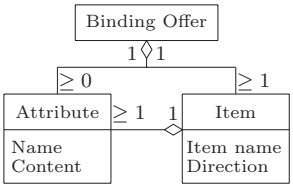

Fig. 2. Class diagram of binding offer

A value system for the assessment of offers consists of two parts: items a business parter wants or needs and items s/he owes for merchandise. Each of these items has an assigned attractiveness which depends on the individual preferences.

Based on this value system and the current state of the binding offer, the requester (within transition $evaluate_Q$) and the responder (within transition $evaluate_P$) decide on acceptance, termination, or modification. For this, the attractiveness of the entire offer must be determined by relating the attractiveness of the items being offered to the attractiveness of the items being traded for. In accordance with Carabelea, we call this ratio negotiation balance (*NB*, cf. [6]):

$$NB := \frac{\textit{Attractiveness of all items offered}}{\textit{Attractiveness of all items being traded for}}$$

In terms of Petri nets, the calculation of *NB* can be implemented within a transition of a Predicate/Transition net (Pr/T net, [9]) as demonstrated in figure 3. Places *Want*, *Merchandize* and *Binding Offer* are assumed to be marked by tuples defining the value system and the current binding offer. Braces around the variables annotated at the arcs symbolize that the transition accesses the entire set of tuples. Within the transition, we use terms of the relational algebra (cf. [7]: natural join, projection, selection and aggregation) to specify the access onto these sets.

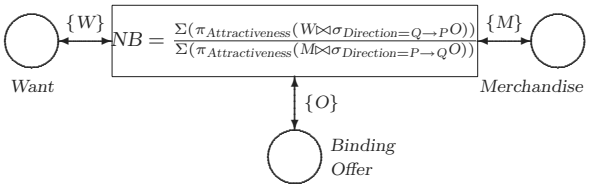

Fig. 3. Net implementation for determining *NB*

For the implementation of the decision making we need to define threshold values. In [6], Carabelea suggests an upper limit for acceptance *ulfa* such that a trader only accepts an offer if $NB < ulfa$ holds true. An upper limit for new proposal *ulfnp* is used to decide on modifying an offer and therefore continuing a negotiation (in the case of $ulfa \leq NB < ulfnp$) or termination ($ulfnp \leq NB$). Based on the Petri net implementation for the calculation of *NB* shown in figure 3, a decision tree can be implemented in terms of a Petri net straight forward as a private element within the entire business process.

Both, the separation of business process elements into private and public as well as the interaction aspect of electronic negotiations demand integration concerning various aspects: the integration of the public processes of two possible negotiation partners in order to find out whether a protocol can be found between them and an integration of the private and public elements of business processes in order to derive the entire model from these fragments. Especially the integration between negotiation partners depends on the existence of a graph exchange language which we discuss next.

IV. GXL for Exchanging Business Process Models

Negotiating business requires to exchange business process models. These business processes enclose at least data on *information*, *data* and *control flow* and/or *organizational structure* [18], [10, p 98ff]. Exchange languages for business process models have to offer an adaptable notation for integrated representation of these views on business process models. The *GXL Graph Exchange Language* [26] offers such a generic exchange format based on graphs. GXL's adaptivity is given by metamodeling technology. GXL metaschemas, which formally define graph classes, specify the concepts of modeling languages and their dependencies. GXL metamodels for exchanging control flow aspects of business processes using Event-driven Process Chains or Petri-nets have been shown in [27] already. Integrated reference schemas for visual languages including schemas for information and organizational structures are given in [25].

Figure 4 shows the GXL-metaschema for the data flow view of business processes modelled by workflow nets as defined in section III.

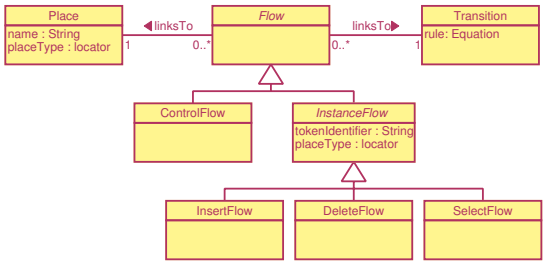

Fig. 4. Data view metamodel

This metaschema defines the ordinary Petri net concepts Place, Flow, and Transition including their connections. Flows are subdivided into subclasses modeling control and instance flows. Additional constraints (not shown here) ensure e. g. the bipartiteness of Petri-nets. Transitions are explained by rules to specify the executed calculation. Places and InstanceFlows refer to data specified within the information view of business process models. Objects of certain types occupy Places or flow between Places and Transitions. These references to information models are modelled by placeType attributes. Their value refers to the corresponding classes in GXL documents exchanging class diagrams describing the information view. Again, additional constraints in the metaschema ensure that Flows incident to Places carry proper tokens.

Within a GXL representation the business process is depicted as a graph according to the domain specific metaschema. GXL documents are exchanged by XML-streams [23]. To express rules, GXL has to be completed by means to describe mathematics. MathML [24] offers an XML language for exchanging mathematical content. GXL provides extensibility by including further XML languages. Thus, GXL can easily be combined with MathML extended by operators of the relational calculus.

We have chosen to use one of the most complex parts of our negotiation model to demonstrate the applicability of our approach. As a private process fragment, it has to be integrated with the public processes of one of the negotiation partners in order to derive a complete model of the negotiation. As mentioned before, also the much simpler process definitions given as WF-nets can be formalized and exchanged correspondingly as has been demonstrated in [27].

Figure 5 shows an extract of a GXL document exchanging the WF-net given in figure 3. The first two lines specify the used XML version and refer to the GXL document type definition [26]. The GXL representation of the WF-net in figure 3 is enclosed in a <graph>-element. Each Place, Transition, and Flow, is represented by an according <node>-Element of appropriate type. <Edge>-elements represent the connections between these nodes. Place `Want` is represented by <node> $p0$ (lines 7-12). Attribute `placeType` (line 10-11) refers to the class definition of Want which is defined in another GXL document `businessObject.gxl`. Lines 14-42 specify (a part of) the Transition in figure 5. The attribute `rule` (line 16) defines the binding of *NB*. The extract of MathML-code (cf. lines 21-34) refers to the denominator of *NB* and uses additional functions

```
1  <?xml version="1.0"?>
2  <!DOCTYPE gxl SYSTEM
3           "http://www.gupro.de/GXL/gxl-1.0.dtd">
4  <gxl xmlns:xlink="http://www.w3.org/1999/xlink">
5  <graph id="NBRule" edgeids="true" edgemode="directed">
6    <type xlink:href="metaPN.gxl#pnSchema"/>
7    <node id="p0">
8      <type xlink:href="pnSchema.gxl#Place"/>
9      <attr name="name"><string>Want</string></attr>
10     <attr name="placeType">
11       <locator xlink:href="businessObjects#Want"/></attr>
12   </node>
13   ...
14   <node id="t0">
15     <type xlink:href="pnSchema.gxl#Transition"/>
16     <attr name="rule">
17       < xmlns="http://www.w3.org/1998/Math/MathML">
18       <apply><eq/>
19         <ci>NB</ci>
20         <apply><divide/>
21           <apply><sum/>
22             <apply><project/>
23               <set><ci>Attractiveness</ci></set>
24               <apply><naturaljoin/>
25                 <ci>W</ci>
26                 <apply><select/>
27                   <apply><eq/>
28                     <ci>Direction</ci>
29                     <csymbol>P&amp;rightarrow;Q</csymbol>
30                   </apply>
31                 ...
32           </apply>
33           <apply><sum/>
34           ...
35           </apply>
36         </apply>
37       <apply>
38       </equation>
39     </attr>
40   </node>
41
42   <node id="f0">
43     <type xlink:href="pnSchema.gxl#SelectFlow"/>
44     <attr name="tokenIdentifier">
45       <string>W</string></attr>
46     <attr name="placeType">
47       <locator xlink:href="businessObjects#Want"/></attr>
48   </node>
49   ...
50   <edge id="e0" from="f0" to="p0">
51     <type xlink:href="pnSchema.gxl#linksTo"/></edge>
52   <edge id="e1" from="f0" to="t0">
53     <type xlink:href="pnSchema.gxl#linksTo"/></edge>
54   ...
55 </graph>
56 </gxl>
```

Fig. 5. GXL representation of Figure 3

<project/>, <naturaljoin/>, and <select/> to denote operations on the according databases. Furthermore, figure 5 shows the GXL representation of the Flow connecting Place `Want` and the Transition (cf. lines 44-50) and the corresponding edges (cf. lines 52-55).

The document in figure 5 shows how the data flow part of a business process is exchanged by GXL. This document refers to another GXL document (cf. line 11) which stores the information view of business processes. The extract of the class diagram in figure 2 is analogously exchanged by an according GXL document using an appropriate GXL metaschema for class diagrams.

V. Integration of the Models

The definition of a cooperative information system requires that the private and public processes of the participating parties can be integrated into each other. Formally speaking, we can interpret the process definition of each (possible) cooperation partner as a specification the other one has to fulfill within the cooperation. A verification whether this holds true or not can be conducted in terms of a Logic of Actions (LoA). Consequently, our focus is on a synchronization of cooperative systems by means of transitions and not by places as has been investigated for WF-nets (cf. [12], [3]).

LoA bases on a definition of *processes* consisting of elementary processes in which actions can *occur* or are *being forbidden sequentially*, *coincidentally* or *independently* from each other. Within a process, the same action may occur repetitively, however actions must not occur and being forbidden in a single process.

Modules, the formulas of LoA, specify sets of processes. Due to alternative process flows not all actions may occur or are being forbidden in every process of a module. Such modules are incomplete and must be *completed* in order to make the module's processes comparable. Completion might also be necessary in between modules especially if modules are considered under the aspect whether they mutually fulfill each other.

Definition 2 (Complete) *Let S be specification and I an implementation formulated as modules over the same set of actions $\mathcal{A}$.*

- *I implies S ($I \vdash S$) iff $\Pi(\mathcal{C}_S[I]) \subseteq \Pi(\mathcal{C}_I[S])$, i.e. the set of processes of I completed with respect to S is a subset of the set of processes of S completed with respect to I.*
- *I is complete with respect to S iff $S \vdash I$.*

Modules of LoA can be implemented in so called *Module nets*. They are transition bounded Petri nets with an explicit *start* transition initially putting tokens onto an empty Module net. Afterwards, the transitions of the net fire as specified by the implemented module until an explicit *goal* transition reproduces the empty initial marking. Auxiliary transitions may be used for structuring the process execution.

As demonstrated by Dehnert (cf. [8, p. 82] there exists a strong relationship between Module nets and WF-nets which allows us to use Module nets for verifying WF-nets against process specifications. Using WF-nets immediately, however, is not possible because of the lack of a negation operator. As a consequence out of this, we use WF-nets for modelling and exchange the WF-net models, verifying, however, is conducted in Module nets automatically derived from the WF-nets.

Proving within LoA bases on considering the conjunction of a specification S and an implementation I. The following theorem proven in [19] points this out:

Theorem 1 (Verification) *For two modules S and I as specified in definition 2, the following holds true:*

$$I \vdash S \Leftrightarrow \Pi(I \boxed{\wedge} S) = \Pi(\mathcal{C}_S[I])$$
$$I \vdash S \Leftrightarrow \Pi(I \boxed{\wedge} \overline{S}) = \emptyset$$

Within Module nets, the module conjunction of LoA is implemented by a so called *join* operator which takes two Module nets as input and joins the transitions pairwise whenever they are sharing the same interpretation (i.e. the *start* transitions, the *goal* transitions, and every two transitions implementing the same elementary process). Auxiliary transitions remain in the result as specified by the input. Also the places are taken as specified in the input. However, we simplify the result and leave away redundant structures whenever possible.

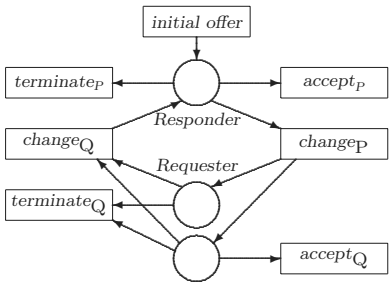

Fig. 6. Module net join of the public processes of requester and responder

Joining the public elements of the processes of the requester and the responder shown in figure 1 results in the net of figure 6. Hereby we observe that the firing of $accept_Q$ never participates in firing sequences reproducing the empty initial marking. So, by applying this approach the requester has the possibility to recognize that he only has the final word in such a negotiation in the case of termination and is now able to decide on accepting a negotiation under this condition or not. For finding cooperation partners, comparable analysis can also be conducted if assumptions concerning the decision making of the negotiation partner is formalized.

VI. Concluding Remarks

Within this paper, we motivate for a differentiation of the elements of business processes into private and public elements. Hereby, the public elements can be used for finding cooperation partners. Such an approach leads to a fragmentation of

business processes (cf. also [20]) which have to be integrated: on the one side the public elements of the processes of two or more cooperation partners are integrated to find a common protocol, on the other side the business partners need to integrate their public and private elements in order to derive a model of the entire business process.

Consequently, we need to formalize our models also appropriate for exchange purposes. We demonstrate the definition of such a format with GXL and explain it by an example. Finally, possible analysis resulting from the possibility to exchange process models are discussed.

Our future work will be on the development of a simulation environment for electronic negotiations. We will base our implementation on a full specification of the exchange objects in GXL.

References

[1] W. M. P. Aalst, van der. Structural Characterizations of Sound Workflow Nets. Computing Science Reports 96/23, Eindhoven University of Technology, 1996.

[2] W. M. P. Aalst, van der. The Application of Petri Nets to Workflow Management. *The Journal of Circuits, Systems and Computers*, 1998.

[3] W. M. P. Aalst, van der and M. Weske. The P2P Approach to Interorganizational Workflows. In K. Dittrich, A. Geppert, and M. Norrie, editors, *Advanced Information System Engineering, CAISE 2001*, volume 2068 of *LNCS*, pages 140–156. Springer, 2001.

[4] B. Baumgarten. *Petri-Netze: Grundlagen und Anwendungen*. BI-Wissenschaftsverlag, Mannheim, Wien, Zürich, 1990.

[5] J. Billington, S. Christensen, K. van Hee, E. Kindler, O. Kummer, L. Petrucci, R. Post, C. Stehno, and M. Weber. The Petri Net Markup Language: Concepts, Technology, and Tools. In *W. van der Aalst, E. Best (Eds.): Applications and Theory of Petri Nets 2003: 24th International Conference, ICATPN 2003, Eindhoven, The Netherlands, June 23-27, 2003. Proceedings, LNCS 2679*, pages 483–505. 2003.

[6] C. Carabelea. Adaptive Agents in Argumentation-Based Negotiation. In Marik, editor, *Conference MASA 2001*, Lecture Notes in Artificial Intelligence 2322, pages 180–187. Springer, 2002.

[7] E. Codd. A Data Base Sublanguage Founded on the Relational Calculus. In *Proceedings of the ACM SIGFIDET Workshop on Data Description, Access, and Control*, November 1971.

[8] J. Dehnert. *A Methodology for Workflow Modeling - From business process modeling towards sound workflow specification*. PhD thesis, TU Berlin, 2003.

[9] H. J. Genrich and K. Lautenbach. System Modelling with High-Level Petri Nets. *Theoretical Computer Science*, 13, 1981.

[10] S. Jablonski, M. Böhm, and W. Schulze, editors. *Workflow-Management, Entwicklung von Anwendungen und Systemen, Facetten einer neuen Technologie*. dpunkt, Heidelberg, 1997.

[11] E. Kindler. On the Semantics of EPCs: A Framework for Resolving the Vicious Circle. In J. Desel, B. Pernici, and M. Weske, editors, *Business Process Management (BPM 2004)*, volume 3080 of *Lecture Notes in Computer Science (LNCS)*, Potsdam, Germany, 2004. Springer.

[12] E. Kindler, A. Martens, and W. Reisig. Interoperatbility of Workflow Applications: Local Criteria for Global Soundness. In v. d. W. M. P. Aalst, J. Desel, and A. Oberweis, editors, *Business Process Management (Models, Techniques, and Empirical Studies)*, volume 1806 of *LNCS*, pages 235–253. Springer, 2000.

[13] K. Lautenbach and C. Simon. Verification in a Logic of Actions. In *7. Workshop Algorithmen und Werkzeuge für Petrinetze*, Koblenz, 2000.

[14] J. Mendling and M. Nüttgens. EPC Markup Language and Refernce Models for XML Model Interchange. draft, 2004.

[15] P. Mutzel, M. Jünger, and S. Leipert, editors. *Graph Drawing, 9th International Symposium, GD 2001 Vienna, Austria, September 23-26, 2001. Revised Papers*, volume 2265 of *LNCS*. Springer, Berlin, 2002.

[16] XML Meta Data Interchange (XMI) Specification. `http://www.omg.org/technology/documents/formal/xmi.htm` (01.09.2001), November 2000.

[17] W. Reisig. *Petri Nets*, volume 4 of *EATCS Monographs on Theoretical Computer Science*. Springer-Verlag, 1985.

[18] A.-W. Scheer. *Business Process Engineering - Reference Models for Industrial Enterprises, 2nd ed.* Springer-Verlag, Berlin, 1994.

[19] C. Simon. *A Logic of Actions and Its Application to the Development of Programmable Controllers*. PhD thesis, Universität Koblenz-Landau, 2001.

[20] C. Simon and J. Dehnert. From Business Process Fragments to Workflow Definitions (accepted). In *EMISA 2004 - Informationssysteme im E-Business und E-Government*, Gesellschaft für Informatik, Luxemburg, 2004.

[21] C. Simon and M. Rebstock. Integration of Multi-attributed Negotiations within Business Processes. In J. Desel, B. Pernici, and M. Weske, editors, *Business Process Management (BPM 2004)*, volume 3080 of *Lecture Notes in Computer Science (LNCS)*, Potsdam, Germany, 2004. Springer.

[22] P. Swatman and P. Swatman. EDI Systems Integration: a Definition and Literature Survey. *Information Society Journal*, 8(3):169–205, 1992.

[23] Extensible Markup Language (XML) 1.0 (Second Edition). `http://www.w3.org/TR/2000/REC-xml-20001006.pdf`, October 2000.

[24] Mathematical Markup Language (MathML) Version 2.0 (2nd Edition). `http://www.w3.org/TR/2003/REC-MathML2-20031021/`, 21 October 2003.

[25] A. Winter. *Referenz-Metaschemata für visuelle Modellierungssprachen*. Deutscher Universitätsverlag, Wiesbaden, 2000.

[26] A. Winter. Exchanging Graphs with GXL. In *[15]*, pages 485–500. 2002.

[27] A. Winter and C. Simon. Exchanging Business Process Models with GXL. In M. Nüttgens and J. Mendling, editors, *XML4BPM 2004, Proceedings of the 1st GI Workshop XML4BPM – XML Interchange Formats for Business Process Management at 7th GI Conference Modellierung 2004, Marburg Germany, March 2004*, pages 103–122, http://wi.wu-wien.ac.at/~mendling/XML4BPM/xml4bpm-2004-proceedings-gxl.pdf, March 2004.