Matchmaking for business processes based on conjunctive finite state automata

Andreas Wombacher*, Peter Fankhauser, Bendick Mahleko and Erich Neuhold
Fraunhofer Gesellschaft,
Integrated Publication and Information Systems Institute,
64293 Darmstadt, Germany
E-mail: andreas.wombacher@ipsi.fraunhofer.de
E-mail: peter.fankhauser@ipsi.fraunhofer.de
E-mail: bendick.mahleko@ispi.fraunhofer.de
E-mail: erich.neuhold@ipsi.fraunhofer.de
*Corresponding author

Abstract: Web services have a potential to enhance B2B e-commerce over the internet by allowing companies and organisations to publish their business processes on service directories where potential trading partners can find them. This can give rise to new business paradigms based on ad-hoc trading relations as companies, particularly small to medium scale, can cheaply and flexibly enter into fruitful contracts, e.g., through subcontracting from big companies. More business process support by the web service infrastructure is however needed before such a paradigm change can materialise. The current infrastructure does not provide sufficient support for searching and matching business processes. We believe that such a service is needed and will enable companies and organisations to establish ad-hoc business relations without relying on manually negotiated frame contracts like RosettaNet PIPs. This paper gives a formal semantics to business process matchmaking and an operational description for matchmaking.

Keywords: matchmaking; service discovery; business process.

Reference to this paper should be made as follows: Wombacher, A., Fankhauser, P., Mahleko, B. and Neuhold, E. (2005) 'Matchmaking for business processes based on conjunctive finite state automata', *Int. J. Business Process Integration and Management*, Vol. 1, No. 1, pp.3–11.

Biographical notes: Andreas Wombacher is a senior researcher working in the area of process integration and service oriented architectures. Peter Fankhauser is also a senior researcher working on data integration. Bendick Mahleko works in the domain of process matchmaking and related indexing structures. Erich Neuhold is the director of the Fraunhofer IPSI.

1 INTRODUCTION

Web services have a potential to enhance B2B e-commerce over the internet by allowing companies and organisations to publish their business processes on service directories where potential trading partners can discover them. This can give rise to new business paradigms based on ad-hoc trading relations as companies, particularly small to medium scale, can cheaply and flexibly enter into fruitful contracts, e.g., through subcontracting from big companies by simply publishing their business processes and the services they offer.

To date, business process descriptions have been mainly investigated as a basis for flexibly coupling processes. This includes approaches for dynamic process composition (Mecella et al., 2001), modelling of cross-organisational

workflows (Kafeza et al., 2001; Aalst, 1999), and support of external process views (Wombacher and Mahleko, 2002; Grefen et al., 2002). But the information published by means of business processes can also significantly facilitate service discovery.

Existing infrastructures to service discovery, fail to utilise this information. UDDI (Ariba et al., 2000), for example, only provides for simple string search.

This is not sufficient for business processes, especially if there do not exist any pre-negotiated and uniquely named frame contracts published by standardisation organisations as, for example, RosettaNet.

Other service based infrastructure face the same issue. In particular, within the ebXML framework business partners can express their business capabilities (including their business processes) using trading partner profiles (TPPs) without providing any means to match these.

This paper presents an approach to more precise service discovery using business process descriptions rather than individual messages.

The next section illustrates limitations of existing approaches to service discovery by way of simple examples of compatible and incompatible business processes. Section Approach formalises a more precise notion of business process matching based on finite state automata. The following section discusses at which level the introduced techniques can be deployed in existing service description frameworks, followed by applying the approach to a real world scenario. The final section concludes and outlines future work.

2 EXAMPLE

Figure 1 depicts two business processes involving two trading parties: a vendor *v* and a customer *c*. Nodes represent the states of a business process; the end states are identified by a double circle. Edges represent state transitions, which are labelled with messages denoted as *from#to#message_name*, where *from* represents the message sender, *to* represents the message recipient, and *message_name* is the name of the message.

Figure 1(a) shows the vendor business process, where the vendor expects to receive a purchase order (*c#v#PO*) message, followed by a credit card payment (*c#v#ccPay*) and finally sends back a delivery (*v#c#Delivery*) message to the respective customer. The customer process depicted in Figure 1(b) also initiates the process with a purchase order request (*c#v#PO*). But then it insists on delivery (*v#c#Delivery*) before payment by credit card (*c#v#ccPay*) or by invoice (*c#v#invoicePay*).

At the level of individual messages these two business processes match. However, because they require a different order of payment and delivery, they are incompatible, that is, they can not successfully interact. In order to avoid matching incompatible business processes we thus need to take into account message sequences rather than individual messages.

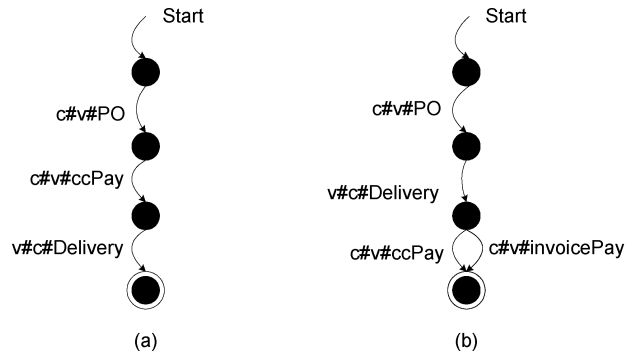

Figure 1 *(a) Vendor message sequence and (b) Customer message sequence*

Figure 2(a) shows a purchase order business process provided by a vendor. The process starts with a purchase order (*c#v#PO*) message, followed by a delivery (*v#c#Delivery*) message, and either a credit card payment (*c#v#ccPay*) or an invoice payment (*c#v#invoicePay*) message. In case the ordered product is not on stock, the vendor may reject a purchase order by sending a no stock available (*v#c#noStock*) message.

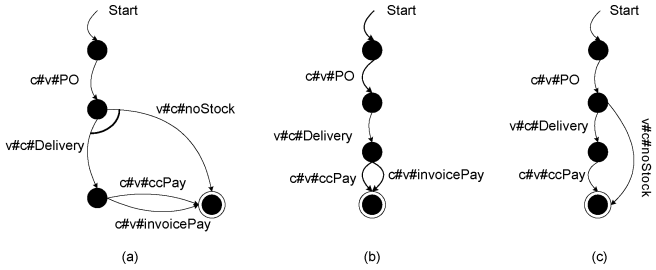

Figure 2 *(a) Vendor message sequence insisting on v#c#noStock and v#c#Delivery messages, (b) Customer message sequence and (c) Customer message sequence with optional v#c#noStock message*

The vendor process involves two kinds of choices. On the one hand, it insists on the availability of both, the *v#c#noStock* and the *v#c#Delivery* message (conjunctive choice, depicted by an arc connecting the two choices). On the other hand, it supports the two payment options as genuine alternatives (disjunctive choice).

Figure 2(b)[1] depicts a customer business process. While this process matches the vendor process with respect to the delivery payment order, it cannot handle the required *v#c#noStock* message. Therefore, the two business processes can not reach the end state, if an ordered product is not on stock.

Conversely, the business process in Figure 2(c) supports *v#c#noStock* and *v#c#Delivery* messages, whereas it supports only one payment option. This process now satisfies all conjunctive choices of the vendor process, and in addition supports at least one genuine alternative of the disjunctive choice. Thus, the vendor process and the customer process are compatible.

In summary, the two examples in Figures 1 and 2 illustrate that

- message sequence
- conjunctive choices need to be taken into account to determine the compatibility of business processes.

3 APPROACH

Finite state automata (Hopcroft et al., 2001) constitute a suitable starting point to model business processes for the purpose of matchmaking. They can represent (possibly infinite) sets of message sequences without explicit modelling of parallel execution capabilities as provided by more expressive approaches such as Petri Nets (Van der Aalst and Kees van Hee, 2002). A similar approach might be applicable to Petri Nets, but for ease of a first evaluation finite state automata (FSA) have been selected. Formally, finite state automata can be represented as follows:

Definition: (Finite State Automata (FSA))

A finite state automaton A is represented as a tuple

$$A = (Q, \Sigma, \delta, q_0, F)$$

where

- Q is a finite set of states
- $\Sigma \subseteq P \times P \times M$ is a finite set of messages in M sent by a sender in P to a receiver in P, where P is a set of all parties being involved
- $\delta \subseteq Q \times \Sigma \times Q$ is a set of transitions
- q_0 a start state with $q_0 \in Q$
- $F \subseteq Q$ a set of final states.

The only difference to the standard definition of FSAs is that the alphabet Σ consists of triples representing messages rather than of atomic tokens. However, for the purpose of matchmaking business processes, these triples can be treated like atomic tokens: Two message triples are equal, if their sender, their receiver, and the message (with its parameters) are equal.

An FSA A generates a language $L(A)$ which enumerates the (possibly infinite) set of all message sequences supported by a business process. The languages for the example business processes depicted in Figure 2 are:

The vendor language of Figure 2(a):

L(vendor) := {c#v#PO -> v#c#noStock,
c#v#PO -> v#c#Delivery -> c#v#ccPay,
c#v#PO -> v#c#Delivery
-> c#v#invoicePay}

the simple customer language of Figure 2(b):

L(customer) := {*c#v#PO -> v#c#Delivery -> c#v#ccPay*,
c#v#PO -> v#c#Delivery
-> c#v#invoicePay}

the customer language with *v#c#noStock* message of Figure 2(c):

$L(customer_{noStock})$:= {*c#v#PO -> v#c#noStock*,
c#v#PO -> v#c#Delivery
-> c#v#ccPay}

Two FSAs match, if their languages have a non-empty intersection. The intersection of two FSAs is again an FSA, which can be determined with the usual cross product construction:

Definition: (Intersection of two FSAs)

The intersection $A_1 \cap A_2$ of two automata $A_1 = (Q_1, \Sigma_1, \delta_1, q_{10}, F_1)$, and $A_2 = (Q_2, \Sigma_2, \delta_2, q_{20}, F_2)$ is $A = (Q, \Sigma, \delta, q_0, F)$, with:

- $Q = Q_1 \times Q_2$,
- $\Sigma = \Sigma_1 \cap \Sigma_2$,
- $\delta = ((q_{11}, q_{21}), \alpha, (q_{12}, q_{22})) \mid (q_{11}, \alpha, q_{12}) \in \delta_1, (q_{21}, \alpha, q_{22}) \in \delta_2\}$
- $q_0 = (q_{10}, q_{20})$, and
- $F = F_1 \times F_2$.

If the resulting automaton does not contain at least one path (possibly of zero length) between the start state and an end state, its language is the empty set Ø. In this case, the business processes modelled by the FSAs are incompatible, because there does not exist any common message sequence. For example, the intersection of the two business processes in Figure 1, does not contain any path between its start and its end state.

The intersection of the vendor process in Figure 2(a) and the customer process in Figure 2(b) is equivalent to the customer process. However, even though this intersection is not empty, it does not contain the required transition *v#c#noStock* of the vendor process, and thus is not a suitable basis for matching the two processes.

The reason for this false match is that standard FSAs can not distinguish between disjunctive and conjunctive choices. Usually, a choice in a standard FSA is regarded as a disjunction. Consequently, the language of an FSA denotes a disjunction of all possible message sequences. However, the intended meaning of the vendor process requires both, disjunction and conjunction. In conjunctive normal form, this meaning can be expressed by the following logical expression on message sequences:

(c#v#PO -> v#c#Delivery -> c#v#ccPay or c#v#PO
-> v#c#Delivery -> c#v#invoicePay)
and c#v#PO -> v#c#noStock

This logical expression can be represented by the conjunction of two FSAs depicted in Figure 3.

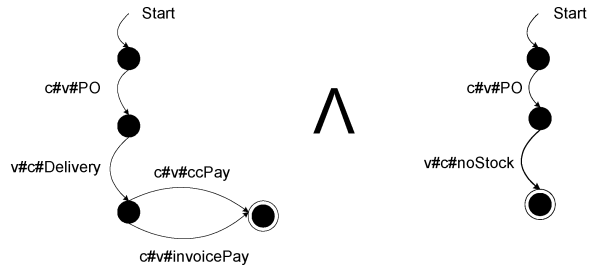

Figure 3 *Vendor message sequence from Figure 2(a) represented as conjunction of disjunctive FSAs*

Business processes represented by two conjunctions of individual FSAs can be matched as follows. Intuitively, two such business processes match, if for each FSA in one business process there exists at least one FSA in the other business process with a non-empty intersection. As can be easily verified, a sufficient condition for this is that the intersection of each individual FSA in one business process with the union of all FSAs in the other business process is non-empty. The following definition formalises this:

Definition: (Intersection of two conjunctive FSAs)

With $C_1 = A_{11}$ and … and A_{1m}, $C_2 = A_{21}$ and … and A_{2n} two conjunctive FSAs, and $C'_1 = A_{11} \cup ... \cup A_{1m}$, $C'_2 = A_{21} \cup ... \cup A_{2n}$ the FSAs resulting from the union of the individual FSAs in C_1, and C_2, the intersection $C_1 \cap C_2$ of two conjunctive FSAs is:

$$\wedge_{1 \le i \le m} (A_{1i} \cap C'_2) \wedge \wedge_{1 \le j \le n} (A_{2j} \cap C'_1)$$

This definition can be easily implemented with an algorithm that is linear in the number of the individual automata A_{1i} and A_{2j}. Note however, that computing the intersection of two (disjunctive) FSAs is quadratic in the number of states. One possible approach to a more scalable algorithm for this intersection may be the deployment of appropriate index structures for regular expressions (e.g., Chan et al., 2003).

If the intersection of two conjunctive FSAs contains an empty automaton, two business processes do not match, because at least one required message sequence of one process is not supported by the other process. Otherwise, they match.

For the compatible business processes depicted in Figures 3 and 2(c) their intersection generates the following logical expression on message sequences (Note that this can be further simplified by discarding the redundant third term in the conjunction):

c#v#PO -> v#c#Delivery -> c#v#ccPay and *c#v#P -> v#c#noStock* and (*c#v#PO -> v#c#Delivery -> c#v#ccPay* or *c#v#PO -> v#c#noStock*)

Because this does not contain the empty choice, the two business processes match.

In contrast, the intersection of the incompatible business processes depicted in Figures 3 and 2(b) contains an empty choice:

(*c#v#PO -> v#c#Delivery -> c#v#ccPay* or *c#v#PO -> v#c#Delivery -> c#v#invoicePay*) and Ø and (*c#v#PO -> v#c#Delivery -> c#v#ccPay* or *c#v#PO -> v#c#Delivery -> c#v#invoicePay*)

Therefore, these two business processes do not match.

As can be easily verified, the (normalised) conjunctive FSAs as introduced above are closed under intersection. However, FSAs, which contain a conjunctive choice at some inner node rather than at the top level, cannot necessarily be transformed into conjunctive normal form.

For example, the simple FSA in Figure 4 cannot be unfolded into conjunctive normal form, because the underlying required set of message sequences is an infinite conjunction of purchase order sequences.

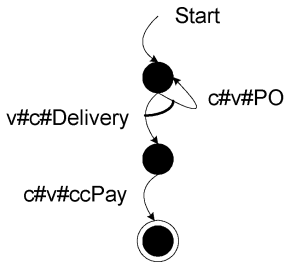

Figure 4 *Example for an infinite conjunction*

Whether and how the intersection of unnormalised conjunctive FSAs can be determined without normalisation is an open issue. Also, the support of explicitly excluded message sequences is an open issue. However, because FSAs are closed under complement, at least support of top level exclusions appears to be straight forward.

4 APPLICATION TO SERVICE DEFINITION LANGUAGES

This section illustrates how the presented approach to matchmaking for business processes can be applied to existing service definition languages. Services are typically described at three major levels: Messages, abstract processes, and execution processes.

Message descriptions such as WSDL and EDIFACT describe the syntax and structure of messages. The Web Service Definition Language (WSDL) (Christensen, 2001) uses XML Schema to describe the input and output of operations supported by a service. These operations can be associated to roles, which correspond to sender and receiver of message descriptions used in this paper. Thus, WSDL descriptions may be used as one concrete form to encode and match individual message descriptions. Alternatively, web based EDI like EDIFACT[2] can be used for this purpose. Such syntactic message descriptions can be matched by component wised comparison. A more ambitious approach is addressed by DAML-S profiles (Ankolekar et al., 2002): these profiles describe messages by means of ontological concepts such that semantic reasoning can be used to more flexibly match messages.

Abstract processes describe the sequences in which messages may be exchanged. There are several proposals for specifying abstract processes, including WSCL, cpXML, the abstract part of BPEL, and ebXML BPSS.

WSCL (Banerji et al., 2002) uses finite state automata to model abstract business processes. As depicted in Figure 5, WSCL descriptions can be easily extended to accommodate top level conjunctions of finite state automata, by introducing a top level element *automataset*, that consists of one or more *conversation* sub-elements.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xsd:id="WSCL" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xsd:targetNamespace="http://www.w3.org/2002/02/wscl10"

 <xsd:element name="Automataset">
  <xsd:complexType>
   <xsd:element name="Conversation" minOccurs="1" maxOccurs="unbounded"/>
  </xsd:complexType>
 </xsd:element>

 <xsd:element name="Conversation">
  <xsd:complexType>
   <xsd:sequence>
     <xsd:element ref="ConversationInteractions"/>
     <xsd:element ref="ConversationTransitions"/>
   </xsd:sequence>
   <xsd:attribute name="name" type="xsd:string" use="required"/>
     .
     .
     .
  </xsd:complexType>
 .
 .
 .
</xsd:schema>
```

Figure 5 *WSCL extension*

Conversation Policy XML (cpXML) (Hanson et al., 2002a, 2002b)[3] extends finite state automata with hierarchical states, which encapsulate again a finite state automaton. Conjunctive hierarchical states also constitute an option to model conjunctive finite state automata.

Thus the techniques presented in this paper can be readily applied to match business processes.

BPEL (Andrews, 2003) (synthesised from XLANG[4] and WSFL (Leymann, 2001)) and ebXML BPSS (Levine, 2001) also allow for the specification of parallel recursive business processes, which can not be described directly by finite state automata. Therefore, for the purpose of matchmaking, parallelism needs to be abstracted away by enumerating all possible paths, which increases the size of the resulting automaton.

Execution process description extends abstract process description with information necessary to execute a business process. This includes the binding of the abstract process to internal processes, constraints on message parameters and on time. This additional information is usually confidential and therefore not advertised publicly. Nevertheless, especially constraints may be deployed for improving the precision of matchmaking.

5 EVALUATION

The application of the proposed approach to service definition languages provides a possibility to improve the quality of service discovery within service oriented architectures. Further, the improved service discovery allows cost reduction in classical EDI based frameworks. The latter one is illustrated in a RosettaNet based B2B marketplace example.

Classical marketplaces are based on a pre-defined global workflow to ensure successful business interaction of the parties using the marketplace. Thus, the marketplace host require the involved trading parties to implement the part of the global workflow they are involved in, that is the corresponding local workflow.

Since EDI integration is expensive, implementing parts of a local workflow that are not relevant to the business for a participating party is not cost effective. Service discovery as introduced within this paper avoids implementing irrelevant parts of the local workflow while keeping the guarantee of successful business interactions.

The B2B marketplace scenario relies on the RosettaNet framework providing commonly accepted message semantics and a clearly specified message syntax as a basis for all workflows implemented in this framework. In addition, RosettaNet provides basic building blocks for a workflow specification called Partner Interface Processes (PIPs) specifying basic bilaterally exchanged message sequences. The example is based on the PIPs 3A4, 3A7, 3A8, and 3A9 related to the RosettaNet order management cluster.

The workflow depicted in Figure 6 uses the same notation as introduced in the Section Approach. The example scenario describes the order management of customer *c* and vendor *v*. A process starts with a customer requesting a purchase order (PO). The vendor confirms the order (*v*:*c*:*POconf* message) if all items could be delivered, or informs the customer about pending or unavailable items by sending an PO update (*v*:*c*:*POupdate*) message. After a customer has received a PO update message he can respond with a PO change request (*c*:*v*:*POchange*) message either confirming or modifying the PO to fulfil his needs, which is followed by a PO change confirmation (*v*:*c*:*POchangeConf*) message sent by the vendor.

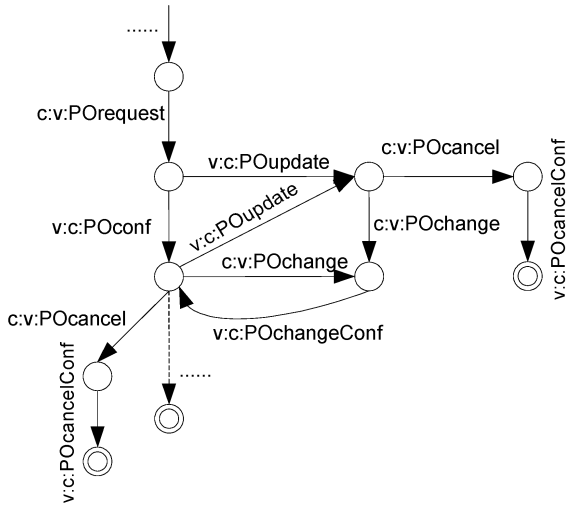

Figure 6 *Marketplace example*

In case the customer receives a PO confirmation, a PO update, or a PO change confirmation message, the customer has the option to cancel the order by sending a *c*:*v*:*POcancel* message, which must be confirmed by the vendor (*v*:*c*:*POcancelConf* message). A cancellation might be necessary because only parts of the offer have been confirmed or the requested delivery date can not guaranteed by the vendor.

The workflow defined above specifies the global workflow. The corresponding local workflow can be

derived by the global one as described, e.g. in Van der Aalst and Weske (2001). In this specific case, the workflow is limited to bilateral interaction, thus the local workflows of the customer and the vendor are equivalent to the global one depicted in Figure 6.

As stated above, neither the customer nor the vendor is willing to support the complete local workflow, because their specific business terms explicitly exclude parts of it.

Within this example, the parties want to reduce the cancel options of their business partner to ensure the trustworthiness of a request or an offer respectively being relevant for example in trading perishable goods like vegetables, fruits, or fresh fish.

Thus, it is assumed that the customer does not allow the vendor to change an order after it has been confirmed by him. The following implemented customer workflow is a subset of the customer local workflow.

Further, it is assumed that the vendor does not allow the customer to cancel an order after he has requested it. The resulting implemented vendor workflow is a subset of the local workflow and depicted in Figure 8.

As stated above, the successful business transaction in classical marketplaces is guaranteed by all parties implementing their pre-defined local workflow completely. The goal is to allow implemented local workflows being a subset of the pre-defined local workflow, but providing the guarantee of successful business interactions to a subset of all potential trading partners. The aim is to find the corresponding trading partners where a successful business interaction can be guaranteed.

A solution is to find the relevant trading partners by using the matchmaking approach presented in this paper. The implemented local workflows are transformed into conjunctive automata and trading is limited to parties where the successful business interaction can be guaranteed, that is, where the conjunctive automata match.

A local workflow represents message sequences that are messages sent and received by the trading party. Usually, the receiving of a message is unconstrained meaning a party must be able to accept a certain set of message types (as specified in the local workflow) without considering the values contained in the message, thus considered as a disjunctive choice. Further, a message sent is selected by a trading party on behalf of its internal state and the already performed message sequence. Thus, the selection of the message is deterministic and a potential trading partner must support the main messages, that is, it is considered as conjunctive choice.

This generic principle has been applied to the implemented local workflow by the customer depicted in Figure 7 resulting in a conjunction of three individual FSAs depicted in Figure 9.

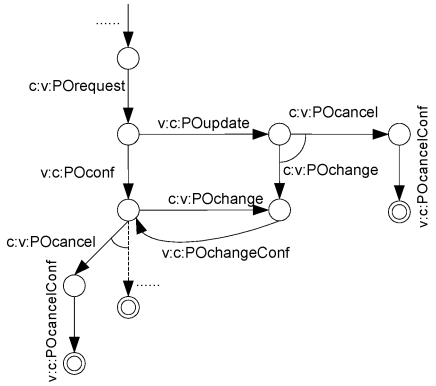

Figure 7 *Implemented customer workflow*

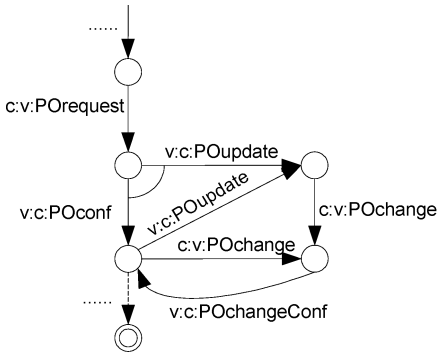

Figure 8 *Implemented vendor workflow*

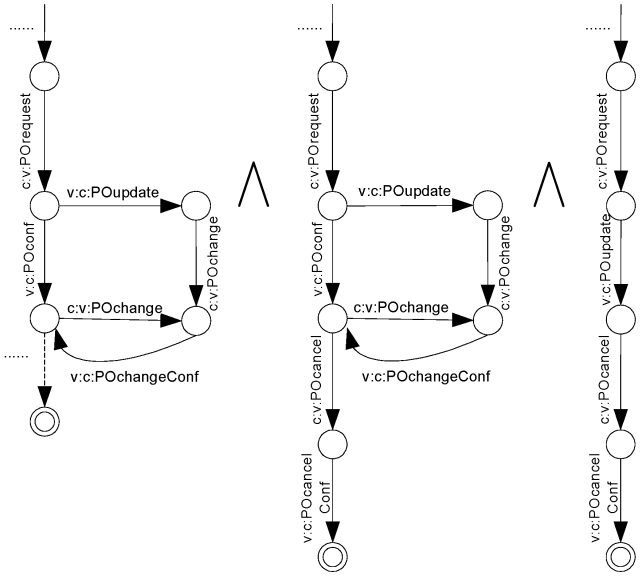

Figure 9 *Customer conjunctive FSA*

Figure 10 depicts the two individual FSAs representing the conjunctive FSA derived from the implemented local workflow of the vendor depicted in Figure 8.

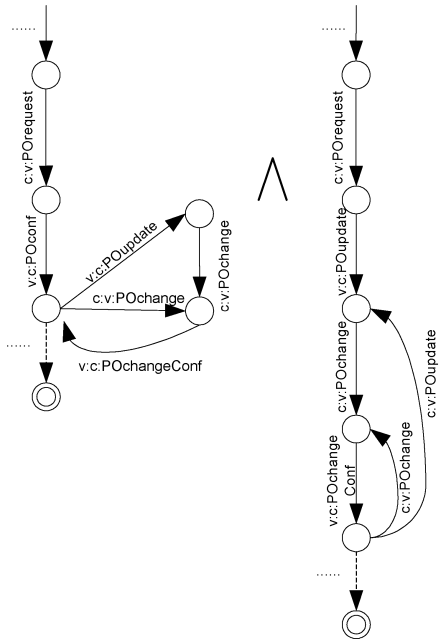

Figure 10 *Vendor conjunctive FSA*

To determine whether the two conjunctive automata match, the intersection is calculated and checked for an empty choice (see Section Approach). The union of the individual FSAs within the conjunctive automata (represented by C'_1 and C'_2 in the intersection definition) is equivalent to the automata depicted in Figure 6 representing the implemented workflow of the customer and the vendor respectively.

In this example, the two conjunctive automata do not match, because the implemented vendor workflow does not support the requested cancel message after a PO confirmation message has been processed, that is an empty choice exists. Thus, the conjunctive automata do not match although the standard FSA intersection of the implemented workflow automata depicted in Figure 6 is not empty.

In this example, the property of mandatory and optional message sequences prevented a false match of potential trading partners, which does not guarantee a successful business interaction.

6 RELATED WORK

Handling processes in inter-organisational cooperation is usually based on the existence of a global predefined workflow split into different parts to be executed locally (Van der Aalst and Weske, 2001; Grefen et al., 2000; Klingemann et al., 1999). Because these approaches are based on a global workflow definition, the local workflows are also known, and thus do not require service discovery considering message sequences. If a collaboration is established without a global pre-defined workflow (Kafeza et al., 2001; Aalst, 1999; Finin et al., 1994; Kimbrough and Moore, 1997), service discovery based on matchmaking message sequences comes into place. Current approaches do not distinguish between conjunctive and disjunctive choices, and thus can only determine whether two processes have at least a single message sequence in common, rather than guaranteeing a successful business interaction between these processes.

Typically, matchmaking must get along with incomplete information (Casati and Discenza, 2001; Georgakopoulos et al., 1999), because trading partners will not publish their business critical information such as the highest price a customer is willing to pay within a negotiation or auction process. The creation of a consistent global workflow from local ones considering this limitation is an open research issue.

Other matchmaking approaches are based on semantic information (Berbers-Lee, 2001; McIlraith, 2001). In Sycara et al. (1999) a language is introduced describing the functional aspects as well as the messages and their parameters based on a domain specific ontology. DAML-S (Ankolekar et al., 2002) uses workflow aspects as well as the functional semantic description of the service within the matchmaking. In contrast to model the complete service description using semantic web technology, the web service offering language (WSOL) provides additional semantic meta-information to increase precision of the service discovery. In particular, classes of services are modelled by specifying functional constraints, Quality of Service (QoS), simple access rights, price, and other constraints in addition to a WSDL description (Tosic et al., 2002). An even more simplified approach (Bernstein and Klein, 2002; Klein and Bernstein, 2001) uses a process ontology to improve precision of key word based querying. The main draw back of semantic annotation is the necessity of a common ontology used for annotating and querying services. Unfortunately, no such ontology currently is in place.

Logic based approaches addressing service discovery are Web Service Request Language (WSRL) and Product Lifecycle Management PLM_{flow}. WSRL (Aiello et al., 2002; Papazoglou et al., 2002) addresses planning of an orchestration and composition of services to fulfil user requirements. While WSRL performs service discovery on behalf of temporal and linear constraints, PLM_{flow} (Zeng et al., 2002) is based on rule inferencing using the specified business rules rather than a fixed workflow. Thus, PLM_{flow} is characterised as a rule-based non-deterministic workflow engine aiming to establish cooperation based on local decidability of the trading partners of their involvement. These approaches are based on the fact that the local workflow model/rules are provided to the trading partners and do not consider the need of hiding business critical information.

The approach presented in this paper is based on modelling message sequences derived from a local workflow model. As stated above, Molina-Jimenez et al. (2003) presents a similar approach, but does not distinguish conjunctive and disjunctive choices. In Mecella et al. (2001) two e-services are compatible if every possible trace in one service has got a compatible one in the second one. Unfortunately, the description of the compatibility check of the traces is not described at all.

7 SUMMARY AND FUTURE WORK

This paper has introduced an approach to match business process descriptions described by means of conjunctive finite state automata. By explicating message sequence and required messages such descriptions allow for more precise matches than current approaches matching only individual messages.

Thereby, existing service description languages based on finite state automata can be readily deployed as a basis for more precise service discovery.

Currently, the approach is being implemented for a business process repository supporting matchmaking and discovery based on business process descriptions. In addition, future work will be devoted to the following issues:

- How can bilateral matching for business processes be extended to multilateral processes? The aim is to investigate to which extend the proposed approach can be applied on multi-party scenarios like for example supply chain example, where potentially process dependencies between the different parties exist.
- How can the syntactic match of individual messages be extended to take into account more semantics?
- How can the rather expensive intersection of finite state automata scale to large service description repositories by means of appropriate index structures? Based on the afore mentioned implementation we will carry out performance testing and performance optimisation.
- How can the intersection of automata be used as a basis for effectively coupling matching processes?

The aim is to come up with constructing multi-lateral collaborations on behalf of existing services each providing its own local workflow. Due to the intersection calculated during the matchmaking message sequences supported by the different parties are calculated, or alternatively, several message sequences have been excluded. The issue is that the coupling of the supported message sequences guarantees successful business interaction.

REFERENCES

Aalst, W.v.d. (1999) 'Interorganizational workflows: an approach based on message sequence charts and petri nets', in *Systems Analysis – Modelling – Simulation*, Vol. 34, No. 3, pp.335–367.

Aiello, M., Papazoglou, M., Yang, J., Pistore, M., Carman, M., Serafini, L. and Traverso, P. (2002) 'A request language for web services based on planning and constraint satisfaction', *Proc. of 3rd Int. Workshop, Technologies for E-Services* (*TES*), LNCS 2444, Springer, pp.76–85.

Andrews, T., Curbera, F., Dholakia, H., Goland, Y., Klein, J., Leymann, F., Liu, K., Roller, D., Smith, D., Thatte, S., Trickovic, I. and Weerawarana, S. (2003) 'Business process execution language for web services', version 1.1, May.

Ankolekar, D-S.C.A., Burstein, M., Hobbs, J.R., Lassila, O., McDermott, D., Martin, D., McIlraith, S.A., Narayanan, S., Paolucci, M., Payne, T. and Sycara, K. (2002) 'DAML-S: Web service description for the semantic web', *Lecture Notes in Computer Science*, Springer, Vol. 2342, pp.348–363.

Ariba, I., Corporation, I. and Corporation, M. (2000) *Universal Description, Discovery and Integration*, September, http://www.uddi.org/.

Banerji, A., Bartolini, C., Beringer, D., Chopella, V., Govindarajan, K., Karp, A., Kuno, H., Lemon, M., Pogossiants, G., Sharma, S. and Williams, S. (2002) *Web Services Conversation Language (WSCL) 1.0 W3C Note*, March.

Berbers-Lee, T., Hendler, J. and Lassila, O. (2001) 'The semantic web', *Scientific America*, Vol. 284, No. 5, pp.34–43.

Bernstein, A. and Klein, M. (2002) 'Discovering services: towards high-precision service retrieval', *Proc. of CAiSE International Workshop, Web Services, E-Business, and the Semantic Web* (*WES*), LNCS 2512, Springer, pp.260–275.

Casati, F. and Discenza, A. (2001) 'Modeling and managing interactions among business processes', *Journal of Systems Integration*, Kluwer Academic Publishers, Vol. 10, No. 2, pp.145–168.

Chan, C., Garofalakis, M. and Rastogi, R. (2003) 'Re-tree: an efficient index structure for regular expressions', *The VLDB Journal*, Springer-Verlag, New York, Inc., Vol. 12, No. 2, pp.102–119.

Christensen, E., Curbera, F., Meredith, G. and Weerawarana, S. (2001) *Web Services Description Language (WSDL) 1.1 W3C Note*, March, http://www.w3.org/TR/wsdl.

Finin, T.W., Fritzson, R., McKay, D. and McEntire, R. (1994) 'KQML as an agent communication language', *CIKM 1994*, ACM Press, pp.456–463.

Georgakopoulos, D., Schuster, H., Cichocki, A. and Baker, D. (1999) 'Managing process and service fusion in virtual enterprises', *Information Systems*, Vol. 24, No. 6, pp.429–456.

Grefen, P., Aberer, K., Hoffner, Y. and Ludwig, H. (2000) 'Crossflow: cross-organizational workflow management in dynamic virtual enterprises', *International Journal of Computer Systems Science & Engineering*, Sep. CRL Publishing, Vol. 15, No. 5, pp.277–290.

Grefen, P., Ludwig, H. and Angelov, S. (2002) 'A framework for e-services: a three-level approach towards process and data management', *Technical Report RC22378* (*W0203-061*), IBM Research Division, Thomas J. Watson Research Center, March.

Hanson, J.E., Nandi, P. and Kumaran, S. (2002a) 'Conversation support for business process integration', *Proceedings 6th IEEE International Enterprise Distributed Object Computing Conference* (*EDOC-2002*), IEEE Press, pp.65–74.

Hanson, J.E., Nandi, P. and Levine, D.W. (2002b) 'Conversation-enabled web services for agents and e-business', *Proceedings of the International Conference on Internet Computing* (*IC-02*), CSREA Press, pp.791–796.

Hopcroft, J.E., Motwani, R. and Ullman, J.D. (2001) *Introduction to Automata Theory, Languages, and Computation*, Addison Wesley.

Kafeza, E., Chiu, D.K.W. and Kafeza, I. (2001) 'View-based contracts in an e-service cross-organizational workflow environment', in Casati, F., Georgakopoulos, D. and Shan, M. (Eds.): *TES 2001 LNCS 2193*, Springer, pp.74–88.

Kimbrough, S.O. and Moore, S.A. (1997) 'On automated message processing in electronic commerce and work support systems: speech act theory and expressive felicity', *ACM Transactions on Information Systems*, Vol. 15, No. 4, pp.321–367.

Klein, M. and Bernstein, A. (2001) 'Searching for services on the semantic web using process ontologies', *Proc. of 1st Semantic Web Working Symposium* (*SWWS-1*), Stanford.

Klingemann, J., Wäsch, J. and Aberer, K. (1999) 'Adaptive outsourcing in cross-organizational workflows', *Proceedings*

of the 11th Conference on Advanced Information Systems Engineering (*Caise*), June, Heidelberg, Germany.

Levine, P., Clark, J., Casanave, C., Kanaskie, K., Harvey, B., Clark, J., Smith, N., Yunker, J. and Riemer, K. (2001) *ebXML Business Process Specification Schema*, May, http://www.ebxml.org/, www.ebxml.org/specs/ebBPSS.pdf.

Leymann, F. (2001) 'Web services flow language (WSFL 1.0)', May.

McIlraith, S., Son, T. and Zeng, H. (2001) 'Semantic web services', *IEEE Intelligent Systems* (*Special Issue on the Semantic Web*), April.

Mecella, M., Pernici, B. and Craca, P. (2001) 'Compatibility of e-services in a cooperative multi-platform environment', in Casati, F., Georgakopoulos, D. and Shan, M. (Eds.): *TES 2001 LNCS 2193*, Springer, pp.44–57.

Molina-Jimenez, C., Shrivastava, S., Solaiman, E. and Warne, J. (2003) 'Contract representation for run-time monitoring and enforcement', *Proc. of Conf. on Electronic Commerce* (*CEC*), IEEE.

Papazoglou, M., Aiello, M., Pistore, M. and Yang, J. (2002) 'Planning for requests against web services', *Bulletin of the Technical Commitee on Data Engineering*, Dec Vol. 25, No. 4, pp.41–46.

Sycara, K.P., Klusch, M., Widoff, S. and Lu, J. (1999) 'Dynamic service matchmaking among agents in open information environments', *SIGMOD Record*, Vol. 28, No. 1, pp.47–53.

Tosic, V., Patel, K. and Pgurek, B. (2002) 'WSOL – web service offering language', *Proc. of CAiSE International Workshop, Web Services, E-Business, and the Semantic Web* (*WES*), LNCS 2512, Springer, pp.57–67.

Van der Aalst, W. and Kees van Hee (2002) *Workflow Management – Models, Methods, and Systems*, MIT Press.

Van der Aalst, W. and Weske, M. (2001) 'The P2P approach to interorganizational workflows', *Proc. of 13. Int. Conf. on Advanced Information Systems Engeneering* (*CAISE'01*)}, *Lecture Notes in Computer Science*, Springer-Verlag, Berlin, Vol. 2068, pp.140–156.

Wombacher, A. and Mahleko, B. (2002) 'Finding trading partners to establish ad-hoc business processes', in Meersman, R. and Tari, Z. (Eds.): *On the Move to Meaningful Internet Systems 2002: CoopIS, DOA, and ODBASE LNCS 2519*, Springer, pp.339–355.

Zeng, L., Flaxer, D., Chang, H. and Jeng, J-J. (2002) 'PLM_{flow} – dynamic business process composition and execution by rule inference', *Proc. of 3rd Int. Workshop, Technologies for E-Services* (*TES*), LNCS 2444, Springer, pp.141–150.

NOTES

[1]This process is equivalent to the one depicted in Figure 1(b) and described above.

[2]EDIFACT *United Nations Directories for Electronic Data Interchange for Administration, Commerce and Transport*, http://www.unece.org/trade/untdid/welcome.htm.

[3]Hanson, J.E. *cpXML: Conversation Policy XML Version 1.0*, http://www.research.ibm.com/convsupport/papers/index.html.

[4]Thatte, S. 'XLANG web services for business process design'.