Modeling Resources in a Service-oriented World

Andreas Heil[1,2]　　　　　　　Martin Gaedke[1]　　　　　　　Johannes Meinecke[1]

[1]Chemnitz University of Technology, Germany
Faculty of Computer Science
{firstname.lastname}@cs.tu-chemnitz.de

[2]Microsoft Research, UK
Computational Science Group
v-aheil@microsoft.com

Abstract

Over the last years, the need to interconnect businesses has significantly affected the Web. The Web has moved constantly from a static source of documents to a dynamic platform for distributed applications. The communication infrastructure of the Web links together applications, e.g. by exposing functionality through Web services in different architectural styles. The current strife between SOA and REST leads one to the issue which approach to choose. Supported by a formal model, we show an integrative way to incorporate service orientation and resource orientation in federated systems as a foundation for future agreements rather than a separation of the approaches.

1. Introduction

The evolution of the Web over the recent years has led to the emergence of modern, Web-based applications, linking different businesses via Web services. Federated portals, or 4th generation portals, are among the recent developments in this field [1]. Service-oriented architectures (SOA) following a WS-* architectural style [2] gained increasing attention, especially in the business to business area. This development was mainly driven by the trend to interconnect multiple enterprises and the need to link their services in order to establish cooperation and federations between these businesses. In the first instance, the technological improvements of Web service influenced this development. The Web service technology stack provides a well developed and standardized set of technologies for creating interoperable service orchestrations, using remote procedure calls, as well as document-driven service interaction [3, 4, 5]. As distributed software components, described on semantic levels, Web services expose functionality and data through well defined interfaces. Standardized Web protocols and formats such as XML, HTTP, SOAP, and WSDL [6] furthermore enable interoperability among a wide range of platforms and allow the deployment of various technologies, best suitable for the particular business need. These vantages can be of value both within as well as outside of an enterprise.

An alternative architectural style, the concept of Representational State Transfer or simply REST [7] led to the development of new applications types based on RESTful Web services [8], mainly in the context of Web 2.0. This development however, led to a irrational discussion about both architectural styles driven by biases and religious arguments rather than by objective facts [9]. REST however, is derived from the constraints given by the Web – constraints, valid for all resources on the Web. Both, SOA and REST must be understood as architectural styles and there is no guarantee that an architecture based on the SOA or REST paradigm does not include Web components based on the concepts of the coexisting paradigms [2]. An example is given in Figure 1 where we see the depiction of a Web site providing information from various content providers, following different architectural paradigms. The question now is whether this composition follows a SOA or a REST style and how to describe relations in a correct manner.
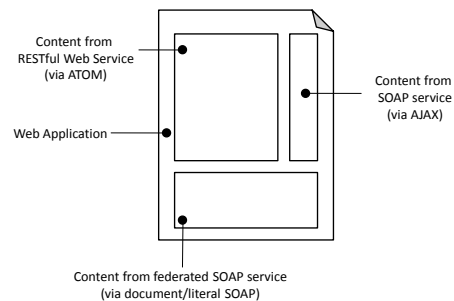

Figure 1. Content from different providers

While there are many attempts to formally describe Web service interaction in SOA-based architectures, only a few try to address this for resource-oriented architectures [10], and it is still an open question how to address both at once.

2. Motivation

To motivate our approach, we consider an example of a credit institution that grants access to its payment service to various online retailers. Each of the partners in such a system might implement its own access control mechanism. In addition, each of the involved partners might be connected to several business partners, who are competitors in their particular field. Hence, instead of granting access to resources to all requestors of a business partner, a fine grained access control is required to secure both internal resources as well as resources made available by third parties. Access to local resources has to be granted based on individual users and services, in order to preserve their autonomy while meeting the partners' business needs. Federated portals or 4th generation portals already address these latest developments. The relations, belonging to multiple organizations, within such federations consist of functionality in the form of Web services, shared data and user accounts. Users, for example, are managed solely by their corresponding organization but can, based on individual trust relationships, access remote resources provided by other organizations. Several challenges have to be address within such a system, including different information, semantics, data and also process models [11, 12]. While functionality of Web applications and services is made available inside and also outside of an organization, complex processes can be established by the composition of building blocks, provided by the various federation partners. On an organizational level, services might be removed, relocated, or substituted but also whole organizations may leave or join a federation. In this paper we describe a methodology to abstract such interactions of system components on the Web, regardless the architectural decision the systems are based on.

3. Modeling Services

Web services, following both, the SOA and the REST paradigm, are not bound to any specific organization or location. Of course, particular services are made available only within some organizational boundaries, accessible only by a limited range of users. Due to their nature, Web-based services and applications, however, tend to be accessible also outside of a specific organization to an audience not necessarily part of this particular organization. The need to federate Web-based applications within the industry finally led to various specifications and technological realizations.

However, all these approaches are based on common logical aspects.

The WebComposition Architecture Model (WAM) is a modeling approach for federated Web-based systems [13, 14, 15] based on these common federation concepts. In WAM, different concerns can be expressed in a multi-layers modeling approach addressing various aspects of the modeled system. Additional layers can be added as extensions, e.g. by third parties, to describe concepts not being part of WAM. Thus, WAM provides inter-model relationships, where the WAM layer is intended to cover the most vital aspects of the federated systems.
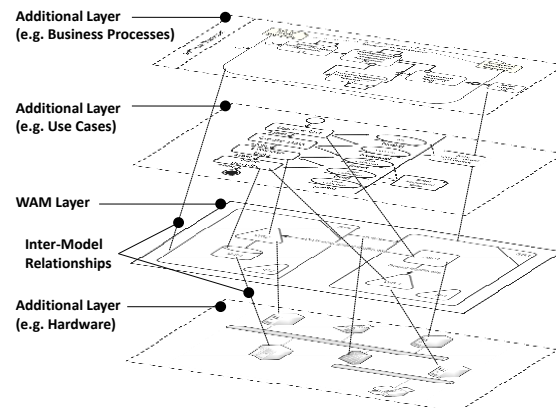

Figure 2. Model layers with inter-model relationships

All model elements in WAM represent entities, which are equivalents of resources, concepts, or relationships linking, various entities including links across model layers such as seen in Figure 2.

The principal WAM modeling elements are constituted in Figure 3. Organizational boundaries, which state a zone of control over Web-based system, network hardware and software systems, are represented as security realms. Each security realm provides a designated security token service (STS). This is a central authority for access control, which provides tokens to access the realm's local resources and services. Authentication requests by unknown users and services are processed by an identity provider (IP). Tokens issued by such an identity provider form the foundation for the STS authorization decision process. The system's components are represented as services that are provided by the different federation partners. Usually, these services are in the form of SOAP or REST architecture-style Web services, while user interaction takes typically place through (Web) applications. WAM allows describing additional resources as further system components connected to services. This includes data providers such as

databases and supplementary legacy systems. If certain aspects of computation are not within the scope of the modeled system, these components can be represented as process units. Potential access to applications and services is indicated by invocation links, which includes invocation within other security realms. Trust relationships can be established between the various security realms by extending the validity of the trusted security realm's security tokens to the trusting realm. To complete the model, the concept of profiles allows the annotation of entities and relationships of the model by labeling them. These labels represent complex communication rules, such as the protocol to use or the encryption algorithms to apply. Therefore, the complexity of communication details can be hidden in a profile database, enabling the reuse of common profiles.
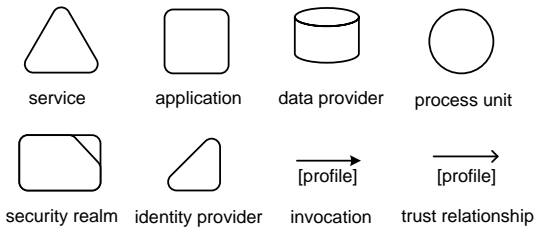

Figure 3. WAM modeling elements

As such, the WAM graphical notation allows designing models in an easy fashion by using pen and paper. To support the modeling and deployment process, we extended Microsoft Visio to provide dedicated support for designing WAM diagrams (cf. Figure 4). A XSL-based transformation engine also allows us to generate machine-readable descriptions of the modeled system, the so-called WAM-XML.
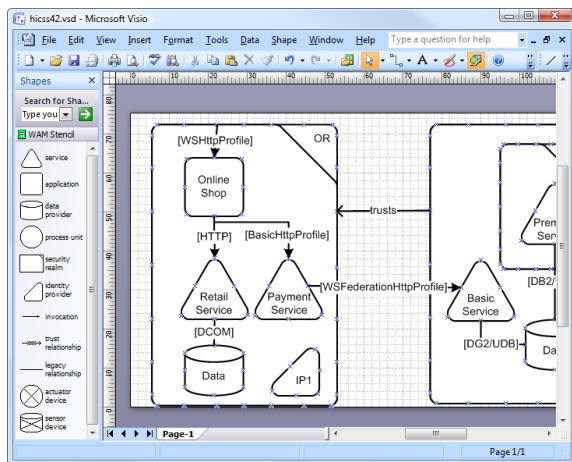

Figure 4. WAM authoring support in Microsoft Visio

To facilitate the implementation of the overall system, a dedicated infrastructure service is provided

for querying and changing the model stored in WAM-XML. Federation partners can use this service to publish their components and relationships. The WAM Service allows the mapping of model modification directly to the infrastructure and vice versa. Hence, it is an up-to-date source for architectural information about the evolving system, its services and relationships.

4. A Formal Approach – State of the Art

Federated Web applications are highly distributed and concurrent systems. In order to choose the right formalism to transpose these models in a correct manner, it is essential to understand the foundations of the used formalisms. Therefore, we discuss the most important theoretical approaches and how they are linked to existing models and methodologies related to the Web.

Each of the formal methods that deal with parallel and concurrent systems, introduced in the next section, provides various advantages for certain application scenarios. Some of them include aspects of security, while some others seem to be similar to concepts in various well known modeling approaches. However, there is an ongoing discussion whether formal methods such as the π-Calculus [16, 17] and Petri-Nets [18] can be applied in a correct manner to existing modeling techniques in the field of Web service composition languages [19]. It is also often questioned if the formalism influenced the modeling technique and if only various aspects of the formalism are recognized in the modeling languages which came casually together without being actually based on the formal methods [20].

4.1. Petri-Net Models

Petri nets have been an early approach in concurrency theory in 1962 [18]. Combining the concepts of states and changes of states, Petri nets allow reasoning about the concurrent behavior of systems. Therefore, a Petri net is defined as n-tuple of places, transitions, flow relations, partial capacity restrictions, a weight function and an initial marking. The default weight function is usually 1, whilst the absence of a capacity restriction is expressed by ∞. The bipartite directed graph spanned by a Petri net then can be described as a 3-tuple of places, flow relations and transitions. A basic capability of Petri nets is their nondeterministic behavior. Any transition satisfying the conditions to be fired can be fired, while none of the transitions has to be fired. This nondeterministic behavior of Petri nets allows us to model basic concurrency behavior. Petri nets have

been under permanent development since initially introduced by Petri. Business modeling languages such as Business Process Execution Language (BPEL) [21], Web Service Flow Language (WSFL) [22] and Business Process Modeling Notation (BPMN) [23] claim Petri nets to be used as theoretical foundation for modeling control flows, using a notion of token-passing within these languages.

4.2. π-Calculus Models

An also often cited foundation for Web service composition in the field of process algebra is the work by Robin Milner on the π-calculus [16, 17]. The π-calculus is argued to be as important for concurrent computation theory as Alonzo Church's λ-calculus [24] for functional programming theory. Both calculi provide the essential mathematical foundation to reason formally about the a system's correctness [25]. The π-Calculus, providing communication capabilities between agents, extended by the concepts of mobility, is an extension of Milner's previous work on the Calculus of Communicating Systems (CCS) [26], where we find the concurrent composition as central element of the calculus, used to synchronize the communication of independent acting agents [26].

Communication between agents is accomplished by exchanging information along named links. Such a link is formed by a negative prefix $\bar{y}x$, describing the output, and a positive prefix $y(z)$ for the input. For agents $P \equiv \bar{y}x.P'$ and $Q \equiv y(z).Q'$ a named link y is established. In other words, P is willing to send x, while Q is ready to receive it along the named link y. Computation within the π-calculus is performed by applying reduction rules in the form of $R \longrightarrow R'$. The π-calculus allows us to reason about logical communication flow between and among agents. However, relocating agents within the communication structure is accomplished by rearranging the communication links between the agents.

Similar to Petri nets, the π-Calculus's foundation is consulted to cover the basics of several modeling approaches. Namely, we can find the Business Process Execution Language for Web Services (BPEL4WS) [27], Business Process Modeling Language (BPML) [28], Web Service for Business Process Design (XLANG) [29] but also WSFL again, claiming some of its foundations in the theoretical approach of the π-Calculus. Also in [10] we find especially an approach using the π-Calculus to describe link-passing within REST-like architectures.

4.3. Mobile Ambients and Variants

As one of the most recent process calculi, the Ambient Calculus [30] was developed by Cardelli and Gordon considering two distinct aspects in mobility: mobile computing and mobile computation. There, mobile computing deals with computation carried out in mobile devices such as laptops, mobile phones and all kinds of ubiquitous devices not bound to any particular location. Mobile computation, however, considers mobile code moving between devices such as applets or agents. The work was inspired by the idea of the World Wide Web, where mobile agents cannot simply migrate from any point A to any point B. For example, to enter or leave administrative domains, explicit authorization is required. The calculus aims to integrate both aspects into one single framework, based on ambients representing the mobile agents, their interaction and mobility.

Similar to interferences known from CCS or the π-Calculus, within Mobile Ambients, plain interferences might appear, causing a non-deterministic behavior of the corresponding program. In addition, Levi and Sangiorgi [31] observed furthermore grave interferences causing not only non-deterministic programs but also leading to eventually program failures and thus categorized by them as programming errors. These issues have been addressed in their work about Safe Ambients by introducing co-actions, coordinating the ambient interactions. Furthermore, ambients, processes as well as capabilities are typed in their approach. In addition to basic types, based on the original ambient calculus, single-threaded as well as immobile types are introduced. Due to the introduction of co-actions, movements become synchronous and thus synchronization mechanisms are required for distributed implementations. Further typing has been introduced by Bugliesi and Castagna in [32], allowing to reason about additional security aspects of Safe Ambients.

Within Mobile Ambients, the unrestricted usage of the open capability, which reveals the content of an ambient to its outside, might lead to unexpected side effects. In practice, this requires incoming code to be statically checked and certified prior to being granted access to local resources and data. Bugliesi et al. addressed this issue with Boxed Ambients in [33]. This variant of the Ambient Calculus drops the open capability, while a finer mechanism for ambient interaction is provided. In Boxed Ambients, communication takes place via anonymous channels. Remote communication between siblings is only possible by applying mobility or intervention of the

parent ambient. Furthermore, communication resources are local to ambients and message exchange results always from explicit read and write request of these resources. Boxed Ambients are also used to reason about access control to resources [34] as well as the design of secure mobile applications using the Channel Ambient Calculus [35]. Based on Mobile Ambients, combined with the capability to use channels, the Channel Ambient Calculus, for example, can be used to model communication in hierarchical network topologies.

5. Inter Model Relationships

When thinking of the Ambient Calculus, mobile ambients are intuitively associated with agents, where an agent can be understood as an executing unit of code. But beyond this initial concept, originating from the agent-based approach, an ambient in the Ambient Calculus is prevailingly based on the existence of boundaries around a particular construct. Thus, ambients are characterized as bounded places where computation happens. More precisely, an ambient (1) has a unique name, (2) provides a collection of agents in the form of threads or processes, (3) can hold a collection of subambients, each with the same characteristics, and (4) can be moved as a whole. This can be a Web page bound by an actual file, a Web Service bound by its executing address range on its hosting system, a database bound by its physical volume or an XML file again bound by the file itself – literally any resource we find on the Web. A virtual construct such as an administrative domain bound by its logical construct can be identified as an ambient in the above meaning as well. This idea for mobile ambients originated basically from the concepts of the Web as state in [30]:

 "The inspiration of this work comes from the potential for mobile computation over the World-Wide Web. The geographical distribution of the Web naturally calls for mobility computation […]. Because of recent advantages in networking and language technology, the basic tenets of mobile computation are now technologically realizable. The high-level software architecture potential, however, is still largely unexplored.".

To show the precise relation between the concepts of the WAM elements introduced in Section 3 and the encoding in the Ambient Calculus we identify the core WAM concepts and their corresponding characteristics as mentioned above.

a) A service, in the form of a SOAP or REST-like architecture style Web service, can be understood as the mechanism providing access to a certain capability, while the service is defined by its interfaces.

b) Similarly, we can identify an application which is bound, for example, by the folder structure on the host system, to which the Web application is deployed to.

c) Data providers, for example databases, are bound by their physical volumes and their logical structures. Also a sensor network, acting as a data provider through a corresponding wrapper, is bound by its physical limitations such as the number of nodes in the sensor networks. Data providers are already recognized as separate entities in the WAM approach. Interfaces to databases might be given through SOAP or RESTful Web services such as the Amazon S3, Amazon SimpleDB [36, 37, 38] or the Microsoft ADO.NET Data Services [39].

d) Additional functionality provided by third party systems, beyond data management, is represented by process units. Process units are recognized as bounded units providing a specified set of functionality or hosting processes out of scope of the modeled system.

e) Security realms are logical constructs, used to envelope services and applications and consequently constitute their boundaries by "self".

f) Invocations do not provide boundaries since they certainly do not represent ambients. However, invocations include the transmission of messages, where sending messages in a web-based system should be understood as the exchange of documents, such as sending a XML file due to a HTTP request using a RESTful service or sending an SOAP message due to a SOAP request. Though, each of these involved files can be recognized as ambient.

6. Modeling Approaches

The constructs identified so far line up well with the ideas based on the characteristic of mobile ambients. Similarly to the π-Calculus, the Ambient Calculus provides a set of basic primitives. These basic primitives have been introduced by Cardelli and Gordon in their original work on the Ambient Calculus. The fundamental difference here is that names n are ambient names and not channel names. This, however, leads us to a fundamental question whether to use channels or not to model federated

Web applications. To point this out, we pick up on the π-Calculus introduction from Section 4.2. Based on this introduction we consider a scenario where multiple services are involved.

In [40] we find an intuitive scenario, which involves a client computer, a printer and a server handing over the communication link after the user request from the user was verified. More in general, we can think of a user who is supposed to use a certain resource (e.g. the Payment Service in our example). The communication link to the resource can be only established after requesting a valid security token from a STS. Both communication links are private to the involved, realized by encryption. This restriction is encoded in the calculus below as well. After the user credentials are approved, the communication link is established and the user gains unqualified access to the resource as illustrated in Figure 5a and b. In this example we clearly see the direction of the communication flow from the STS to the user. This is based on the fact, that the request is send first to the IP and then forwarded to the STS, which finally issues the security token to the user. In a strictly SOA-based system, these invocations can be clearly identified as service invocations between the user's application, the Web service and the STS.
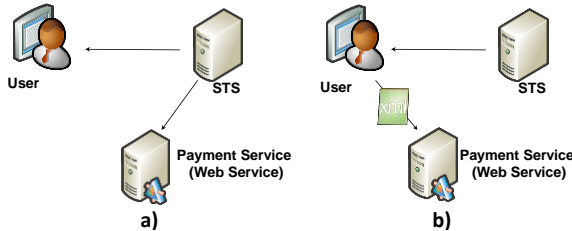

Figure 5. Depiction of a SOA-based resource access using a security token service

To express this scenario as π-Calculus model we need three resources P, Q, R. Communication links are established between resource P and Q as well as between Q and R, depicted as a flow graph in Figure 6 a). Now we want to reconfigure the communication links, letting P directly communicate to R. Therefore, we let Q send its link to R to P. In the π-Calculus we write $Q \equiv \bar{x}y. Q'$. In addition, P has to receive the link expressed by $P \equiv x(z). \bar{y}z''. P'$. $x(z)$ indicates that the variable z is bound within P'; $\bar{y}z''. P'$, hence expresses the will of P to send the information z'' along a link y before it continues as P'. In this example the restrictions (x) and (y) are used to indicate the private nature of the communication links between P, Q and R. The system described so far is expressed in the calculus by

$$(x)(y)(\bar{x}y. Q' \mid x(z). \bar{y}z''. P' \mid y(z'). R').$$

After executing the communication steps between P and Q, the communication link y is established between P and R. Figure 6 b) depicts the expression right before the information z'' is sent.
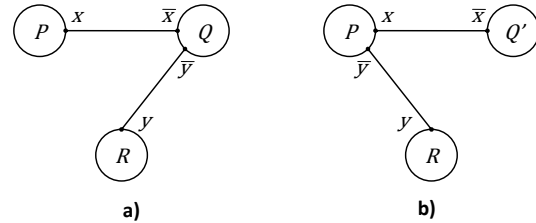

a)　　　　　b)

Figure 6. Basic π-Calculus communication links of the corresponding resource access in Figure 5

Due to the lack of semantics, however, it is hard to say if the model above describes the logical communication flow between the agents or their hierarchical alignments. Therefore, the rearrangement of communication links could be also understood as equivalent to the hierarchical rearrangement of agents. In a REST-based scenario the representations of the resources sent among the services are the central aspect, not the invocation of the service itself.

To address this, we use the Ambient Calculus to clearly distinguish between the location of an ambient and the related communication flow and the resources represented by this communication flow. Since ambients can be nested, we can model hierarchical assembled systems representing labeled, unordered trees. That way we can model the semantics of the system of interest [41] and clearly decide if a service in the model is part of a security realm (cf. Figure 7), what boundaries are passed by an invocation and what resources are involved in this particular invocation.
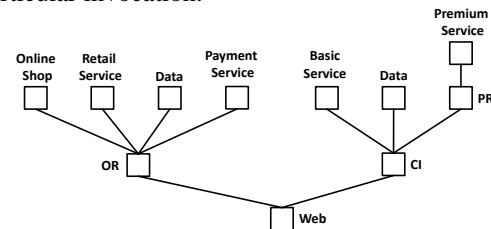

Figure 7. Hierarchical order of ambients

In terms of communication, the Ambient Calculus, allows to encode other formalisms like the λ-Calculus or the π-Calculus using communication primitives. These communication primitives can be used to encode invocations between services as used

for the π-Calculus example. However, also applications and further entities within and among security realms can be modeled, while the interaction among the involved resource can be modeled with a finer granularity. Direct communication between processes takes place in the form of Ambient I/O. As inter-process communication, it takes place between various constructs within a security realm. Inter-ambient communication as provided by Parent I/O or Ether I/O, provides messages-based communication where messages are sent from one process to another, passing the surrounding ambient boundaries. Parent I/O and Ether I/O can thus be used to encode invocations among nested security realms or realms located both within a third security realm. Remote I/O provides functionality for long-range communication. Within the calculus, this type of communication is modeled as so-called messenger agent movements across administrative boundaries. Ether I/O allows communication between any two anonymous ambients through their surrounding ambients ether. In Table 1 we provide a brief overview of the given communication mechanisms.

Table 1. Ambient-based communication

Mechanism	Scope	Type
Ambient I/O	Local	Within ambients, anonymous
Parent I/O	Spatial	Between anonymous parent and named child
Ether I/O	Global	Between named siblings
Remote I/O	Global	Between named ambients

7. Linking Business Process and Federation Models

Based on the concepts introduced so far, we want to provide a brief insight into the WAM's capability of linking various models and how the proposed formalism helps to achieve this goal. Therefore, Figure 8 depicts a business process for an online order through an online retailer's (OR) Web application. The activities *check product availability* and *process order* are accomplished by different services. A further party involved in this process is a credit institution (CI) providing a payment service. Within the activity *process payment*, OR can send the request to two different services provided by CI. Based on the provided endpoint descriptions, both services are appropriate. To distinguish, a further cost-based function is considered where a single transaction is either charged with 3% and a fixed rate of $0.10 or 5% and a fixed rate of $0.25. While

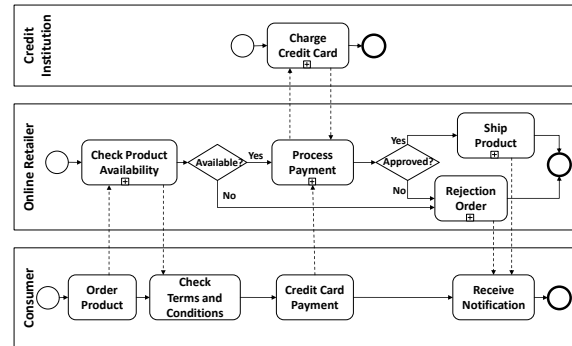initially using the more expensive service, the process is changed to use the obviously cheaper one.

Figure 8. Business process involving multiple parties

Based on such a BPMN notation, executable BPEL can be generated [42] to support an engineering process. However, after changing the business process we realize that the corresponding activity cannot be completed. While the business process model appears to be correct, the corresponding WAM diagram shows that the related Premium Service is not accessible due to a missing trust relationship to the particular service. Access to the related service is only granted to business partners providing additional fee. The inaccessibility of this premium service, however, is not obvious in the BPMN model or the generated BPEL but the restriction to this premium service can be determined due to the additional security realm (PR) in the corresponding WAM model in Figure 9.
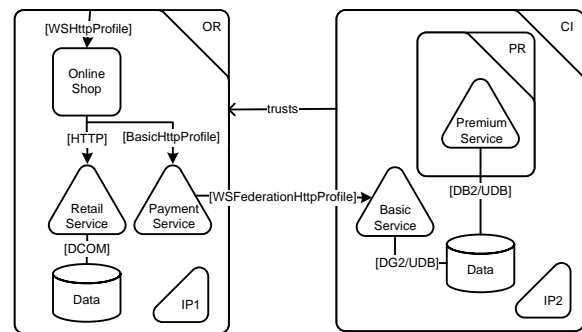

Figure 9. WAM model involving multiple parties

Changes in the federation can have immediate effect on the business process as well. Since the system topology is not necessarily reflected in the business process, changes in the topology are hard to address. By changing business partners, existing trust relationships become obsolete and new ones are established. The related business processes are thus affected immediately by such changes. By changing the credit institution, the *process payment* activity

might become obsolete. Due to the inter-model mapping between the WAM layer and the Business Process layer, these changes are recognized and can be applied to related entities in other modeling layers. Based on [43] we can now identify the various communication links and resources within the model for further, automated computation (cf. Figure 10).
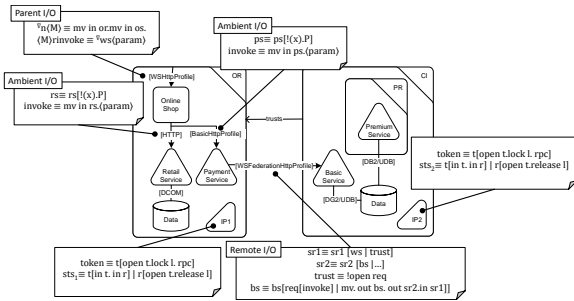

Figure 10. Formalized resources and invocations based on the WAM diagram

The formalized models allow us to monitor and predict potential problems in the various model layers. For predictions, vital elements of the model can be modified to compute potential effects on other system components. For example, by removing a service from the model to simulate the failure of this particular service or changing the corresponding invocation profile, possible effects on the business model can be shown. Various scenarios can thus be modeled either to validate the system's rigidity or to determine weakness in the system.

8. Conclusion

In this paper we discussed the formal foundation of various formal modeling approaches for Web-based systems. We have argued about the advantages and disadvantages of various approaches to show that a strict separation of Service-oriented Architectures based on a WS-* architecture and REST-like architecture styles is not always feasible. With the WebComposition Architecture Model we provide a modeling approach for designs and proofs that can be easily initiated by the user, by using the provided editor for Microsoft Visio, while possible problems are pointed out in the WAM diagram. The complexity of the formal evaluation is thus taken away from the user's burden, who finally receives the important information in a human readable and easy to understand form.

WAM related tools, examples and additional information are available for download http://www.WebComposition.net/WAM.

9. References

[1] D. Gootzit and G. Phifer, "Gen-4 Portal Functionality: From Unification to Federation", Gartner, Stamford, CT, USA, SPA-20-7217, 2003.

[2] A. Bradley, N. Gall, and R. W. Schulte, "Understanding and Applying the Design Differences Between WS-* Based Architecture and Web-Oriented Architecture", G00147780, 2007.

[3] S. Weerawarana, F. Curbera, F. Leymann, T. Storey, and D. F. Ferguson, Web Services Platform Architecture: Prentice Hall International, 2005.

[4] D. Kossmann and F. Leymann, "Web Services", Informatik Spektrum, vol. 27, pp. 117-128, 04 March 2004.

[5] K. Ballinger, D. Ehnebuske, C. Ferris, M. Gudgin, C. K. Liu, M. Nottingham, and P. Yendluri, "WS-I Basic Profile Version 1.1", http://www.ws-i.org/Profiles/BasicProfile-1.1-2004-08-24.html (11-05-2007).

[6] F. Curbera, M. Duftler, R. Khalaf, W. Nagy, N. Mukhi, and S. Weerawarana, "Unraveling the Web services web: an introduction to SOAP, WSDL, and UDDI", IEEE Internet Computing, vol. 6, pp. 86-93, March-April 2002.

[7] R. Fielding, "Architectural Styles and the Design of Network-based Software Architectures", University of California, Irvine, 2000.

[8] L. Richardson and S. Ruby, RESTful Web Services: O'Reilly, 2007.

[9] C. Pautasso, O. Zimmermann, and F. Leymann, "RESTful Web Services vs. "Big" Web Services: Making the Right Architectural Decision ", 17th International World Wide Web Conference, Bejing, China, 2008.

[10] H. Overdick, "The Resource-Oriented Architecture", IEEE Congress on Services (Services 2007), Salt Lake City, UT, USA, 2007, pp. 340-347.

[11] E. J. Barkmeyer, A. B. Feeney, P. Denno, D. W. Flater, D. E. Libes, M. P. Steves, and E. K. Wallace, "Concepts for Automating Systems Integration", National Institute of Standards and Technology, NISTIR 6928, 2003.

[12] D. Libes, D. Flater, E. Wallace, M. Steves, A. B. Feeney, and E. Barkmeyer, "The Challenges of Automated Methods for Integrating Systems", International Conference on Software Engineering (ISASTED), Innsbruck, Austria, 2004, pp. 485-492.

[13] J. Meinecke and M. Gaedke, "Modeling Federations of Web Applications with WAM", Third Latin American Web Congress (LA-WEB 2005), Buenos Aires, Argentina, 2005, pp. 23-31.

[14] J. Meinecke, M. Gaedke, F. Majer, and A. Brändle, "Capturing the Essentials of Federated Systems", Fourteenth International World Wide Web Conference (WWW)*,* Edinburgh, UK, 2006.

[15] J. Meinecke, M. Gaedke, F. Majer, and A. Brändle, "Modeling and Managing Federated Web-based Systems", 3rd International Conference on Web Information Systems and Technologies (WEBST)*,* Barcelona, Spain, 2007, pp. 15-22.

[16] R. Milner, J. Parrow, and D. Walker, "A Calculus of Mobile Processes, Part I", Information and Computation*,* vol. 100, pp. 1-40, Sept 1992.

[17] R. Milner, J. Parrow, and D. Walker, "A Calculus of Mobile Processes, Part II", Information and Computation*,* vol. 100, pp. 41-77, Sept 1992.

[18] C. A. Petri, "Kommunikation mit Automaten", University of Bonn, Bonn, 1962.

[19] W. M. P. v. d. Aalst, M. Dumas, and A. H. M. t. Hofstede, "Web Service Composition Languages: Old Wine in New Bottles?" Euromicro Conference (EUROMICRO'03), 2003, pp. 298 - 305

[20] W. M. P. v. d. Aalst, "Pi calculus versus Petri nets: Let us eat "humble pie" rather than furhter inflate the "Pi hype"".

[21] A. Alves, A. Arkin, S. Askary, C. Barreto, B. Bloch, F. Curbera, M. Ford, Y. Goland, A. Guízar, N. Kartha, C. K. Liu, R. Khalaf, D. König, M. Marin, V. Mehta, S. Thatte, D. v. d. Rijn, P. Yendluri, and A. Yiu, "Web Services Business Process Execution Language Version 2.0", http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html (10-30-2007).

[22] F. Leymann, "Web Services Flow Language 1.0", 2001.

[23] OMG, "Business Process Modeling Notation Specification", 2006.

[24] A. Church, The Calculi of Lambda-Conversion, Reprint 2nd Print ed. New York: Kraus Repr, 1965.

[25] R. Milner, "Elements of Interaction: Turing award Lecture", Communications of the ACM vol. 36, pp. 78-89, Januar 1993.

[26] R. Milner, A Calculus of Communicating Systems vol. 92/1980. Berlin, Heidelberg: Springer, 1980.

[27] T. Andrews, F. Curbera, H. Dholakia, Y. Goland, J. Klein, F. Leymann, K. Liu, D. Roller, D. Smith, S. Thatte, I. Trickovic, and S. Weerawarana, "Business Process Execution Language for Web Services Version 1.1", 2003.

[28] A. Arkin, "Business Process Modeling Language", 2002.

[29] S. Thatte, "XLANG Web Services for Business Process Design", 2001.

[30] L. Cardelli and A. D. Gordon, "Mobile Ambients", First International Conference on Foundations of Software Science and Computation Structures (FoSSaCS '98) at ETAPS'98*,* Lissabon, Portugal, 1998, pp. 140-155.

[31] F. Levi and D. Sangiorgi, "Controlling Interference in Ambients", 27th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*,* Boston, Masachusets, USA, 2000, pp. 352-364.

[32] M. Bugliesi and G. Castagna, "Secure Safe Ambients", ACM SIGPLAN Notices vol. 36, pp. 222-235, March 2001.

[33] M. Bugliesi, G. Castagna, and C. Silvia, "Boxed Ambients", Lecture Notes in Computer Science vol. 2215 / 2001, pp. 38-63, 2002.

[34] M. Bugliesi, G. Castagna, and S. Crafa, "Reasoning about Security in Mobile Ambients ", CONCUR 2001 - Concurrency Theory: 12th International Conference*,* Aalborg, Denmark, 2001, pp. 102-120.

[35] A. Phillips, "Specifying and Implementing Secure Mobile Applications in the Channel Ambient System", University of London, London, 2005.

[36] Amazon Web Services LLC, "Amazon Simple Storage Service Developer Guide", http://docs.amazonwebservices.com/AmazonS3/2006-03-01/ (06-03-2008).

[37] Amazon Web Services LLC, "Amazon SimpleDB Developer Guide", 2008.

[38] F. Shanahan, Amazon.com Mashups. Birmingham, UK: Wrox Press Ltd., 2007.

[39] P. Castro, "Project Astoria", The Architecture Journal*,* pp. 12-17, 2007.

[40] J. Parrow, "An Introduction to the π-Calculus", in Handbook of Process Algebra, J. A. Bergstra, A. Ponse, and S. A. Smolka, Eds. Amsterdam: Elsevier Science, 2001.

[41] A. Heil and M. Gaedke, "Environment-Awareness: Quantitative Processing of Context Changes", 5th IEEE Workshop on Context Modeling and Reasoning (CoMoRea) at the 6th IEEE International Conference on Pervasive Computing and Communication (PerCom'08)*,* Hong Kong, 2008.

[42] S. A. White, "Using BPMN to Model a BPEL Process", 2005.

[43] A. Heil, M. Gaedke, and J. Meinecke, "Identifying Security Aspects in Web-based Federations", IEEE International Conference on Web Services (ICWS 2008), Beijing, China, 2008.