A Robust Business Resource Management Framework Based on a Peer-to-Peer Infrastructure

Thomas Friese[1,2], Jörg P. Müller[1]
[1]Siemens AG Corporate Technology
Intelligent Autonomous Systems
Otto-Hahn-Ring 6
D-81739 München, Germany
joerg.p.mueller@siemens.com

Matthew Smith[2], Bernd Freisleben[2]
[2]Dept. of Mathematics and Computer Science
University of Marburg
Hans-Meerwein-Str
D-35032 Marburg, Germany
{friese,matthew,freisleb}@informatik.uni-marburg.de

Abstract

In this paper, we present the design and implementation of a business resource management framework (BRMF) that allows to manage various business resources on top of a peer-to-peer infrastructure which is highly scalable and resilient against node failure. The BRMF is geared towards the management of static business resources such as business documents, web service descriptions and business process specifications involved in business interactions among various partners. Additionally, it offers the possibility to incorporate management functionality for the control of active business resources such as web services and business process execution engines. The BRMF is a peer-to-peer based middleware solution supporting the development of intra- or cross-organizational business applications. It has been used to implement an application that supports business process execution engines by realizing additional fault tolerance based on a distributed service registry.

1 Introduction

Electronic collaboration of networked enterprises is an important issue for every company to be competitive in the future. Different resources are involved in a typical business application that spans organizational boundaries. Access to certain business applications and internal processes is nowadays often provided through web service interfaces. Web services are used to pass business documents between different enterprises or between departments of an organization. The internal and external processes that govern the interaction of all involved parties are captured in business process specifications that in turn are executed by specialized business process execution engines.

The growing number of business processes that are executed by complex information and communication technology systems creates an increased dependence on those systems. Coupled with the fact that the configuration and development of business applications has become more complex and demanding as more applications and subsystems are integrated, a new approach for the integration of the business domain and the system management environment is needed.

In this paper, we present the business resource management framework (BRMF) as such a new approach. It offers facilites for the storage and transmission of various static resources involved in business interactions as well as the management of active components. It uses a peer-to-peer (P2P) system as a fundamental component that offers an information infrastructure for the management of various resources. The P2P paradigm provides a system design that is inherently failure resistant and self-organizing, thus reducing the management tasks required by the users of such a system.

The paper is organized as follows. In section 2, a sample application is presented. Requirements for the business resource management framework are derived in section 3. The design of the BRMF is presented in section 4. An overview of the implementation of the BRMF is given in section 5. A brief description of the realization of the sample application based on the BRMF is discussed in section 6. Related work is presented in section 7. Section 8 concludes the paper and outlines areas for future research.

2 A Motivating Sample Application

To motivate our work, we refer to a collaborative product design (CPD) process in the automotive sector (see figure 1). In the early design and strategic sourcing phase for a new product (e.g. a new model of a car), an Original Equip-

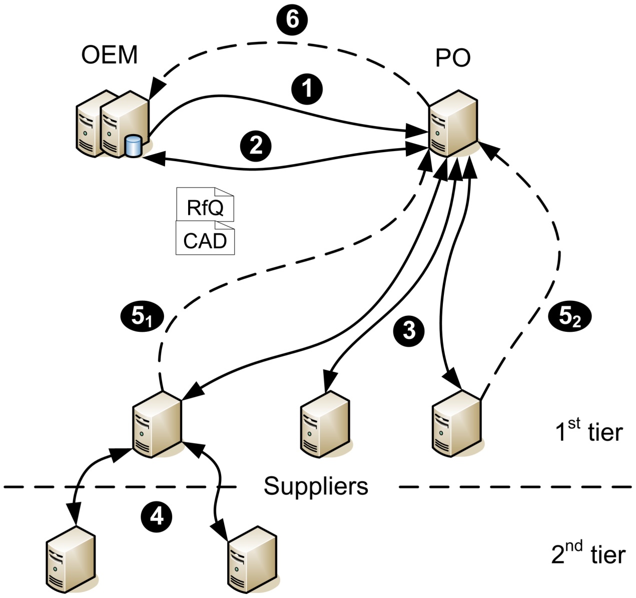

Figure 1. CPD Process: The OEM issues a RfQ (1) that is discussed among OEM and PO (2). Subsequently, PO and Suppliers negotiate the RfQ (3) possibly involving 2nd tier suppliers (4). Offers are collected by the PO (5.1,5.2) and relayed to the OEM (6).

ment Manufacturer (OEM) involves potential suppliers for parts (such as the gear box) to comment on the technical requirements and economic conditions of a possible contract. We consider a simplified process where the OEM issues a request for quotation (RfQ) that is processed and commented upon by a purchasing organisation (PO) of the OEM - which can be a division of the OEM or an external organization. The finalized RfQ is then published to a selection of possible first tier suppliers. Suppliers use internal processes and may involve second tier suppliers to process the RfQ and comment on the technical specifications or economic requirements bound to the RfQ. Some of the attachments can be collected in the RfQ object itself, other parts will be specified by reference to some external documents, such as Microsoft Word files or CAD drawings containing administrative or technical descriptions. After a negotiation phase, the suppliers will come up with individual offers that are relayed to the OEM after collection and preselection by the PO.

3 Requirements for the BRMF

A central requirement for the BRMF is the ability to manage business objects. With the term *business object* we refer to a collection of properties exchanged during business interactions. Often, these objects will adhere to a schema that can be defined and expressed using XML Schema or an interface definition language.

As an example, consider the CPD process presented in the previous section. The BRMF should provide an easy to use interface for the application developer that allows to create and manipulate a business object such as the RfQ, while hiding the complexity of the underlying communication infrastructure from the originator of the object as well as any receiving party that uses or updates the object. Functions must be offered to *publish, retrieve* and *alter* the object.

The CPD scenario is a very sensitive phase in the interaction among different enterprises. The same can be expected for many other business interactions that involve the exchange of business documents. Confidentiality and security of the information enclosed is a great concern for the publisher of a document and all other parties involved. On the one hand, it might be feasible to publicly invite suppliers to bid, on the other hand there may be scenarios in which one supplier is explicitly invited to process a request for quotation or a similar document. Also, responses to an RfQ from different suppliers must be kept confidential since the existence of response could influence subsequent offers.

Therefore, the system has to provide mechanisms to cover both cases, allowing to publish a document - probably based upon the topic of the document - as well as directly transmit the document to a specific recipient. Additional mechanisms for encryption or access control to certain parts of an object and subsystems of the infrastructure may be implemented to achieve confidentiality of the objects managed in the system. Again, the complexity of the underlying security infrastructure should be shielded from the application developer by an easy to use abstraction mechanism.

The business resources covered so far are static in the sense that they do not contain any internal logic to process requests or otherwise produce data actively. Examples for such active business resources (or simply actors in the business application environment) are web services and Process Execution Engines. Business processes are often complex web services themselves that are composed of basic web services acting as the building blocks of the more complex business processes. The de facto standard for the expression of the application flow of such a business process is the Business Process Execution Language for web services (BPEL4WS) [8]. In contrast to the explicit specification of the process flow, a business process may also be governed by the external behavior resulting from the internal actions of all partners.

A basic requirement for the use of active resources is the discovery of their availability. The external interface of a web service is typically described using the web service Description Language (WSDL) [3]. With the ability to manage WSDL documents, the BRMF meets the requirement

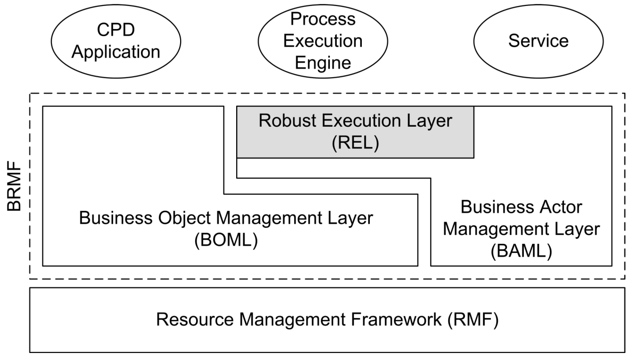
CPD Application
Process Execution Engine
Service
BRMF
Robust Execution Layer (REL)
Business Object Management Layer (BOML)
Business Actor Management Layer (BAML)
Resource Management Framework (RMF)

Figure 2. Layers of the Business Resource Management Framework

to provide the functionality of a web service registry, similarly the process specification of a business process can be managed in the system. Apart from the static description of a web service or business process, dynamic state information for the service or process should be accessible through the system. Examples are internal fault conditions or the complete failure of a web service and also the state of the execution of a single instance of a business process. The availability of monitoring data or state information may ultimately be used to also modify the behaviour of a web service or a process execution engine or autonomously counter heavy load situations through service replication.

4 Design of the BRMF

We now present a layered approach for the design of the BRMF that is depicted in figure 2. The BRMF is built on top of the RMF [5] that offers message based communication between all nodes in the P2P network, fault tolerant publishing, discovery and retrieval of general XML resources and an additional publish/subscribe mechanism. The layer built directly on top of the RMF is the Business Object Management Layer (BOML) that provides an easy way to manage passive business objects on top of the underlying P2P infrastructure. The Business Actor Management Layer (BAML) implements functionality to monitor and control active components of business applications such as web services, Process Execution Engines and business processes. Every additional layer makes use of the functionality of the directly underlying layer. The Robust Execution Layer (REL) provides support for the fault tolerant execution of business processes.

4.1 Representation and Management of Business Objects

As stated before, we refer to collections of properties that adhere to some schema specification as business objects. For many of those objects, an XML serialized form will exist. Even though the basic entity that is managed by the RMF is an XML resource, it is not possible to publish arbitrary XML documents right into the RMF. The following two possible ways of publishing the document in the RMF exist:

- If the object is an XML document or can be serialized as an XML document it is possible to construct an RMF resource as an envelope containing the business object. This resource document has all the required elements needed by the RMF to manage the resource (i.e. the unique resource id and a set of keywords associated with the resource) in addition to the original business object XML document.

- The second option is to publish a reference resource into the RMF instead of the entire business object as an XML document. This resource contains the necessary information to discover the object and retrieve it from an external object management system.

In either case it is essential to choose an appropriate set of keywords that can be used to publish the resource (i.e. the business object) in the P2P information space. The choice of the keywords is vital for later discovery of the resource and the scalability of the underlying P2P infrastructure. A wrong choice of keywords may lead to a degeneration of the information space if, for example, every resource maps to the same peer address. The appropriate mapping from object properties to keywords associated with the corresponding RMF resources is generated using the binding compiler depicted in figure 3.

The Business Object Binding Compiler takes the schema information for a specific kind of business object in conjunction with an object specific binding profile as input and generates all the necessary artifacts to manage business objects of this specific kind in the P2P network.

Defining the mapping from the schema of a business object to a business resource that can be managed in the underlying P2P infrastructure is a non-trivial task. The user is supported in this task by a profile editor. This profile editor is used to select a number of properties from the schema specification and determine the mapping of these properties into properties of the business resource that should be created by the binding compiler. The most important mapping to be supported by the profile editor is the selection of business object properties that should be mapped into keywords of the resulting resource.

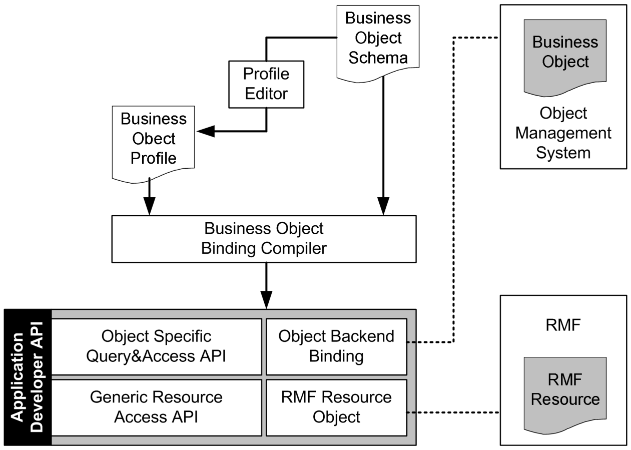

Figure 3. Creating the Business Object Binding for the BRMF.

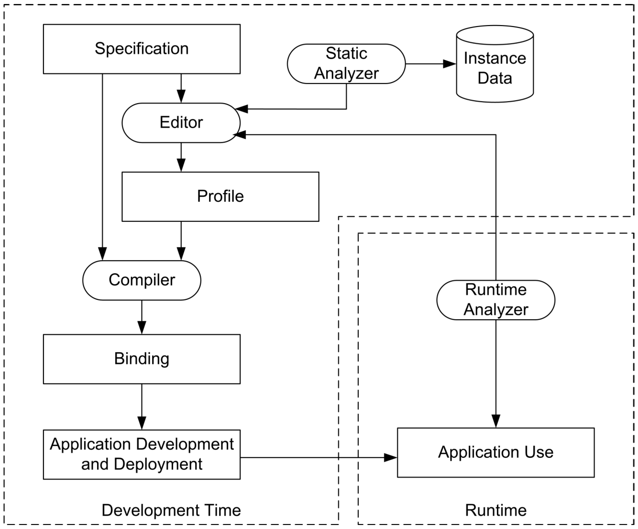

Figure 4. Development and Usage Cycle for the Business Resource Bindings.

The application development cycle is depicted in figure 4. Starting from a schema specification of the objects to be managed, the profile editor is used to construct P2P specific binding information. Both the schema specification and the binding profile are used by the binding compiler to generate a language specific binding (i.e. a class structure that can be instantiated to create and manage business resources of the specific type). The resulting binding is used for the development of business applications.

We envision two possible approaches to optimize the selection of resource properties as keywords. If instance data is available (i.e. a collection of XML documents of a specific schema type), the editor may analyze the data corresponding to a schema element the user is trying to select as a keyword and warn the user if the selection could lead to the degeneration of the P2P information space.

This development time tool may be supplemented by a dynamic optimization mechanism that monitors the instance data managed during the runtime of the applications based on the resource binding. This tool is capable of generating warnings for the user (i.e. the binding developer) to alter the selection of the keywords and ultimately the generated binding.

In addition to the keyword selection information, the resource profile contains information about object specific access methods in order to be able to generate the appropriate access and query API for the application developer. If the objects themselves are stored in an Object Management System such as a document management system or a concurrent version system instead of directly in the RMF, appropriate access methods must be generated to be able to retrieve the object content from those systems. The input and output of the binding compiler naturally depends on the schema format and the target language. The generated backend binding depends on the object management system used to store the business objects. The generated application developer interface contains generic object access methods that allow the developer to work with raw objects (i.e. retrieval of the object such as a CAD drawing as byte stream) while taking care of basic RMF and object management functions. In addition to these generic methods, a business object specific binding may be included in the generated object binding to allow a more convenient or sophisticated way of access to the properties of the business object (i.e. retrieval of a port type object in the host language created from the underlying WSDL specification).

In the design of the BOML we distinguish between three general strategies to maintain business resource information in the P2P information space. The changes to a business resource may be published in the P2P information space either upon explicit request of the resource holder or automatically when the change occurs in the resource (through the property accessor methods of the resource bindings). As a third strategy, the resource binding may collect changes to the properties of a resource and wait for the occurrence of subsequent changes before publishing the changes in order to minimize the network load resulting from frequent changes to resource properties.

4.2 Management of Active Components

While the BOML is intended for the management of static business resources, the management of active components (actors for short) in the business application domain (i.e. web services and business process execution engines) is the functionality provided by the *Business Actor Management Layer* (BAML). The BAML consists of components offering both end user management applications and APIs (e.g. for business process execution engine developers).

The requirement of actor discovery is met by using the functionality of the underlying BOML. A corresponding resource binding for the static descriptions of the actors can be generated and exposed via a specialized interface. The BAML would in this way provide functionality to register services using their WSDL interface descriptions to an application developer. If an end user is not an application developer himself, a generic management application should expose the corresponding functionality. For example, a service provider can use a generic service registration application that takes the URIs of WSDL service descriptions as input and registers the services for use in the BRMF.

More sophisticated functionality is provided by specialized components collected in the BAML. For instance, the *Robust Execution Layer* (REL) [6] offers self-healing execution of business processes without requiring a change to either the process specification or the process execution engine. In the light of a service failure, the REL uses the service lookup functionality of the BAML to locate alternative invocation targets and redirects the service call.

In a similar fashion, the BAML can expose an interface to tunnel service invocations through the underlying P2P communication infrastructure, offering the ability to implement a load balancing and scheduling system in the BAML.

5 Realization of the BRMF

As mentioned in the previous chapter, the BRMF uses a layered approach consisting of two separate layers that are built upon an underlying DHT based P2P infrastructure. In the following we will present some of the details concerning the realization of the separate layers. We will describe our system bottom up, starting with the P2P infrastructure and continuing with the BOML and BAML.

5.1 The Resource Management Framework - A Peer-to-Peer Infrastructure

Our realization of the BRMF is based on the *Resource Management Framework* (RMF) [5]. The RMF is a P2P system that collects a number of nodes to provide a DHT called the *Information Space* of the RMF. Basic operations that can be performed on this logical information space are the publishing and retrieval of data elements as well as the search for such elements in a synchronous as well as an asynchronous way. Additionally, a client can request to be notified whenever a data element matching a certain query is published, removed or otherwise changed in the system. The RMF uses leasing and replication of data elements to ensure persistence of the published information even under conditions of node failure. Data objects in the information space are referred to as *RMF resources*. They are XML elements that have the following child elements:

- The mandatory ID of a resource that is used to uniquely identify the element. This can either be a globally unique UUID or some fully qualified hierarchical name, guaranteeing uniqueness of the ID.

- An optional name of the resource to be used as a user friendly name for application independent presentation of the resource.

- An optional list of keywords to be associated with the resource.

- Any number of application specific XML elements.

Developers are free to determine the values of the root element as well as its namespace specification. The system can still handle a resource if it contains the mandatory resource ID from the RMF namespace as a child element.

When a resource is published in the RMF information space, an internal mapping to peer addresses in the system is calculated to get a list of nodes that are ultimately used to store the resource. This calculation is based on the ID and keywords associated with a resource. The search operation works in two phases: First, the same mapping calculation as for the publishing of a resource is used to determine peer addresses for a set of given keywords or resource IDs. Afterwards, a query is performed on the local share of the information space held by this set of peers. The query is expressed using the XPath query language [14] and yields every matching resource in the local part of the information space as a result.

5.2 The Business Object Management Layer

The BOML is concerned with the management of static business resources in a P2P network. In order to provide this functionality to the application developer in a natural and easy to use manner, it has to generate a representation of the business resource. Furthermore, it offers the functionality to store and manage the resource representations in the underlying P2P information space generating and exposing a resource specific high level API. The BOML uses the resource management facilities of the RMF to manage business resources. Therefore, the BOML business resource binding contains a mapping of a business resource into a RMF resource.

The RfQ in figure 5 will be used to illustrate the tools and mechanisms implemented in the BOML. This RfQ contains a request of the OEM for the production and sourcing of certain parts within a specified time. In addition to the internal elements of the RfQ, this document contains a reference to an external technical specification that is stored in an external document management system.

```
...
<rfq xmlns="http://www.fb12.de/ns/cpd">
 <partName></partName>
 <partID></partID>
 <quantity></quantity>
 <timeframe></timeframe>
  ...
 <techspec
  <!-- external CAD document -->
  xmlns:xlink =
   "http://www.w3.org/1999/xlink"
  xlink:type="extended">
  <dms_locator
   xlink:type="locator"
   xlink:href=
        "http://www.fb12.de/dms/ts_13.cad"/>
  <connector_module
    brmf:moduleID=
    "7dc13b38-acee-4731-9292-9a20a845a290"
    brmf:moduleName=
    "RMF DMS Backend Connector Impl"/>
 </techspec>
  ...
 <originator>
  <name/>
   ...
 </originator>
  ...
</rfq>
```

Figure 5. Request for Quotation Sample Document.

For the generation of the resource binding, the developer opens the XML schema definition for the RfQ document in the profile editor of the BOML. The profile editor allows the selection of document elements or attributes as keywords in the resulting RMF resource. Furthermore, the business resource elements to be directly published in the RMF information space can be specified as well as a restriction on the retrievable elements. Thus, only certain elements can be used for search queries or can be retrieved from the originator of the resource respectively. The generated RMF resource document may contain the entire content of the business resource or just the selected elements and an endpoint reference allowing the system to retrieve the entire resource.

If instance documents adhering to the current schema definition are available, the static analyzer evaluates the suitability of the chosen keyword candidates with respect to the later performance of the search and retrieval operations in the P2P information space. If, for instance, the variance in the value of a keyword occuring in the instance data set is too low, the system marks the keyword as unsuitable for information distribution in a DHT based P2P system.

The resource binding generation is a two step process: First, the binding compiler of the Castor Project [4] is used to generate a standard Java binding for the business resource schema. We have extended the Castor implementation to generate a modified java binding hierarchy that incorporates certain additional elements, such as cascading property value change listeners generating events for property changes that in turn can be processed by our wrapping resource code. In a second phase, a wrapper class is generated that incorporates the Castor binding hierarchy and adds a management implementation to seamlessly integrate the business resource into the RMF system. The representation of the business resource is now contained within the specialized RMF resource class that enables both access to the schema specific properties of such a resource and the generic management functions to be used with the RMF.

The object constructed from the specialized RMF resource class contains a reference to the responsible RMF registrar instance for that node. It also exposes functionality to subscribe to notifications on resource changes within the contained business resource representation as well as the state of the wrapper object.

The binding also contains generated code to manage resource elements that are stored externally and only linked into the resource. Such an external link is shown in figure 5. Here, a technical specification is linked into the document including the identification of a backend connector module. These modules are needed to retrieve the document remotely. Using the XLink locator element, the backend connector retrieves the raw data stream from the remote system and converts into a typed object. The modules themselves are OSGi bundles that can be inserted and updated dynamically in the BRMF middleware, thus allowing the local system to handle new types of objects and remote object management systems. A conceptual overview of the backend access mechanism is shown in figure 6.

A set of retrieval methods of the form `getResourceBy<Element Name>` is generated corresponding to the searchable elements. Additionally, a generic XPath query interface is offered to allow for more complex queries over the entire searchable information portion of the business resource. The application developer can use this business resource binding in his own applications to create and publish resources locally as well as to retrieve and alter resources based on the resource identity or XPath queries.

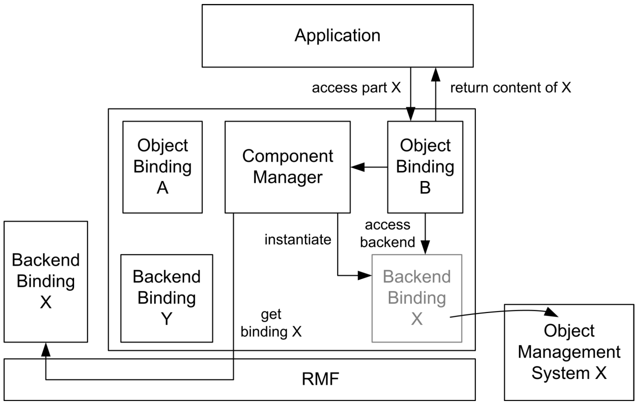

Figure 6. Dynamic Installation of the Backend Binding for Access to Business Resource Elements that are Stored in an External Object Management System.

5.3 The Business Actor Management Layer

A RMF binding for WSDL descriptions has been generated similar to the RfQ example shown in the previous section. A `ServiceRegistrar` has been implemented in the BAML using this WSDL resource binding. This `ServiceRegistrar` offers API functions to build applications that register web services with the RMF information space based on a WSDL specification. It also contains a GUI application and a registration daemon that can be used by service providers to configure a list of services (based on WSDL file locations or arbitrary URI specifications) that in turn is used by the registration daemon to publish the service descriptions and monitor the availability of the service.

The robust execution layer is implemented as a HTTPServlet that acts as a transparent, configurable add-on to any BPEL execution engine to support the self-healing execution of business processes that are managed by the engine. The REL intercepts service invocations of the engine and decides weather to redirect the call to another service or directly try to invoke the service (more details on the decision process can be found in [6]). If the service invocation fails, the REL uses the WSDL resource binding provided by the BOML in order to locate alternative invocation target services and transparently redirects the call.

Figure 7 shows how the functionality of the REL is realized in the BRMF. The REL works as a transparent add-on to a standard process execution engine for BPEL4WS process specifications (BPE^2). It is assumed that the service providers use the `ServiceRegistrar` in the BAML to publish their services (1). This registration is performed using the WSDL resource binding of the BOML (2). A service call emitted by the process execution engine (3) is intercepted by the REL (4). If the invocation of the target service (5) results in an error, the REL uses the `ServiceRegistrar` (6) to locate alternative target services for invocation (7). The result of this invocation is then returned to the process execution engine (8).

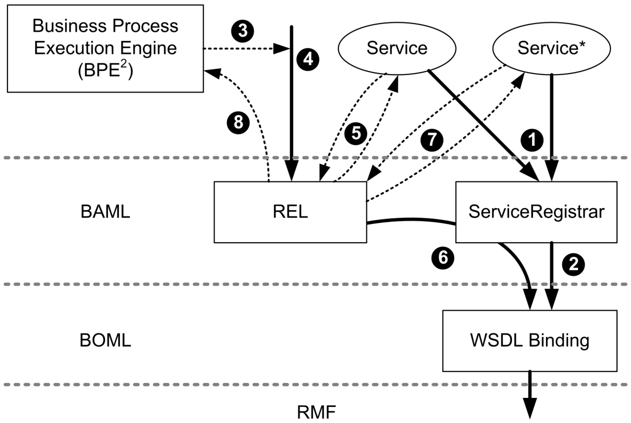

Figure 7. Component Interactions to Realize the REL Component of the BRMF.

6 Realization of the CPD Example

The first step towards the realization of the CPD example presented in section 2 from a developers perspective is the construction of the resource binding for the RfQ documents. Starting from a given XML schema specification for the RfQ contents, the developer uses our binding editor to select keywords and defines the binding profile. Then, the binding compiler is used to generate the resource implementation for the RfQ binding, containing access, query and security functionality for the management of RfQ resources in the BRMF. Then, the OEM, PU and Supplier applications supporting the CPD process are built using the generated resource bindings. Since the keyword mappings for the RfQ resources are integrated into the bindings, subscription and search for RfQ resources is trivial, significantly reducing the complexity and time for the application development.

From the user perspective, the OEM application is used to generate the RfQ resource and to publish it into the P2P information space. The CPD application of the PU has either registered for notification upon availability of RfQ resources or performs a synchronous search after the RfQ is published. The binding utilized by the PU allows it to modify certain elements of the RfQ. The OEM is informed about these changes via the notification mechanism. During this time, the suppliers are not able to access the RfQ document. The security mechanisms preventing unauthorized access are integrated into the bindings and the BOML and RMF layers (however, the details of the security mechanisms are outside the scope this paper). Once the PU and OEM have agreed on a final version of the RfQ, the PU broadens the visibility of the RfQ to the suppliers. Selection of the suppliers to be recipients of the RfQ is based on previous interaction or the discovery of suppliers that published their contact details using a supplier description schema that was handled similarly to the RfQ schema.

Based on the contact information for the OEM that is encoded in the RfQ document, the suppliers send offers using their CPD application.

7 Related Work

The basis for our BRMF is a resilient P2P resource management infrastructure. In a very rough categorization, the spectrum of P2P infrastructures can be divided into two groups. One group uses some form of flooding mechanism to relay requests for information to all nodes in an attempt to discover resources provided by those nodes. Examples for this technology are the gnutella network [15] and the JXTA open source project initiated by Sun Microsystems [7]. While the flooding approach offers a flexible way to formulate information queries, it has some drawbacks regarding the network traffic that is generated by a single search request [10]. Additionally, a negative request for a certain piece of information is no guarantee that the information is not available. Instead, the peer holding the information could have been out of reach for the query which usually has a limited lifetime and outreach to keep the system scalable.

The other group of middleware realizes a distributed information space that is very similar to a distributed hash table (DHT). More strict query semantics are offered by DHT systems where a query can be assumed to yield a result if and only if a certain piece of information matching the query is stored among the nodes forming the DHT. Such a definitive answer is required for business applications built on top of such an information infrastructure. Therefore, we selected a DHT based fabric as the foundation of the BRMF.

An information space offering DHT characteristics is realized by such projects as Chord [13], Pastry [11] or Tapestry [16]. Those systems allow placing and retrieving objects based on the key associated with the object, but they do not support complex XPath queries over the contents of stored XML fragments. RMF goes beyond those basic DHT mechanisms and offers functionality for the storage, discovery, retrieval and monitoring (through subscription and notification mechanisms) of arbitrary XML resources.

To the best of our knowledge, there is no unified system combining business resource management and P2P data management, even though the lack of an intermediary *hub* for business to business communication is an argument for the adoption of a P2P model [1]. The PBiz model [2] extends the Chord routing mechanism to map XML resource descriptions onto a P2P infrastructure. There is no mapping of existing schema definitions into the PBiz system. Business users operate directly on the P2P data model instead, relying on a generic query interface. Also, no details are given on how the system should counter degeneration of the P2P index structure. A model to support business processes in a P2P marketplace scenario is presented in [12]. This work focuses on defining the meta model for market place based interaction without handling resources or a concrete realization of such a system. In [9] a layered business object oriented architecture is introduced supporting meta-modeling, object management, workflow management, directory services and communications. However, only a high level description of the required subsystems based on a a centralized registry model is given.

8 Conclusions

The contributions of this paper are twofold. First, we analyzed the requirements of a business resource management framework built on top of a peer-to-peer information infrastructure with DHT characteristics. Then, we presented the design and implementation of our business resource management framework consisting of two separate components, BOML and BAML, built on top of the RMF - a peer-to-peer infrastructure that is highly scalable and resilient. Great emphasis was placed on the provision of suitable developer tools to enable the easy integration of our BRMF into existing as well as newly constructed business applications. These tools support the developer throughout the entire application development and maintenance process by providing monitoring and analysis tools that generate advice for the redesign of certain application components.

In this paper, we did not deal with the security requirements posed by a distributed business resource managment framework. Consequently, future work will consist of analyzing existing security mechanisms in the field of business resource management and designing and implementating an integrated security solution which will be added seamlessly and transparently into our BRMF. In order to offer better developer support, we will extend the existing tools and test the automation mechanisms and recommendation functionality provided by the P2P keyword mapping analysis. We intend to characterize the management overhead imposed by the different BRMF layers. Finally, it is planned to use the framework for complex real world applications especially in the field of the automotive industry.

9 Acknowledgments

Part of the work reported in this paper is funded by the E.C. within the ATHENA IP under the European grant FP6-IST-507849. The paper does not represent the view of the E.C. nor that of other consortium members, and the authors are responsible for the paper's content.

References

[1] C. Bussler. P2P in B2BI. In *Proceedings of the 35th Annual Hawaii International Conference on System Sciences*, page 302, Big Island, Hawaii, USA, 2002.

[2] S. Chen, Z. Wu, W. Zhang, and F. Ma. PBiz: An E-business Model Based on Peer-to-Peer Network. In *Proceedings of Grid and Cooperative Computing, Second International Workshop*, pages 404–411, Shanghai, China, 2003.

[3] E. Christensen, F. Curbera, G. Meredith, and S. Weerawarana. Web Services Description Language (WSDL) 1.1, 2001. `http://www.w3.org/TR/wsdl`.

[4] Exolab. The Castor Project. `http://www.castor.org`.

[5] T. Friese, B. Freisleben, S. Rusitschka, and A. Southall. A Framework for Resource Management in Peer-to-Peer Networks. In *Proceedings of NetObjectDays 2002*, volume 2591 of *Lecture Notes in Computer Science*, pages 4–21, Erfurt, Germany, 2002. Springer.

[6] T. Friese, J. P. Müller, and B. Freisleben. Self-Healing Execution of Business Processes Based on a Peer-to-Peer Service Architecture. In *Proceedings of the 18th International Conference on Architecture of Computing Systems*, Innsbruck, Austria, 2005. Springer.

[7] L. Gong. JXTA: A Network Programming Environment. *IEEE Internet Computing*, 5:88–95, 2001.

[8] IBM. BPEL4WS: Business Process Execution Language for Web Services, 2003. `http://www-106.ibm.com/developerworks/webservices/library/ws-bpel/`.

[9] A. Karakaxas, V. Zografos, and B. Karakostas. A Business Object Oriented Layered Enterprise Architecture. In *Proceedings of the 11th International Workshop on Database and Expert Systems Applications*, page 807, Greenwich, London, UK, 2000.

[10] J. Ritter. Why Gnutella Can't Scale. No, Really, 2001. `http://www.darkridge.com/~jpr5/doc/gnutella.html`.

[11] A. Rowstron and P. Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In *Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, pages 329–250, Heidelberg, Germany, 2001.

[12] M. Schmees. Distributed digital commerce. In *Proceedings of the 5th international conference on Electronic commerce*, pages 131–137. ACM Press, 2003.

[13] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications. In *Proceedings of the ACM SIGCOMM '01 Conference*, San Diego, California, 2001.

[14] W3C Recommendation. XML Path Language (XPath), 1999. `http://www.w3.org/TR/xpath`.

[15] Wikipedia. Entry on Gnutella. `http://en.wikipedia.org/wiki/Gnutella`.

[16] B. Y. Zhao, J. Kubiatowicz, and A. Joseph. Tapestry: An Infrastructure for Fault-tolerant Wide-area Location and Routing. Technical report, UCB/CSD-01-1141, 2001.