Majer, F., Nussbaumer M., Gaedke M.: A Descriptive Approach for the Lifecycle Support of Distributed Web-based Systems. In Proceedings of 4th International Conference on Web Information Systems and Technologies (WEBIST-2008), 4-7 May, 2008, Funchal, Portugal.

A DESCRIPTIVE APPROACH FOR THE LIFECYCLE SUPPORT OF DISTRIBUTED WEB-BASED SYSTEMS

Frederic Majer, Martin Nussbaumer
Institute of Telematics, University of Karlsruhe (TH), Engesserstr. 4, 76128 Karlsruhe, Germany
{majer, nussbaumer}@tm.uni-karlsruhe.de

Martin Gaedke
Distributed and Self-organizing Computer Systems Group, Chemnitz University of Technology,
Straße der Nationen 62, 09111 Chemnitz, Germany
gaedke@cs.tu-chemnitz.de

Keywords: Web Engineering, Distributed Systems, Web Applications, Web Services, Information Model.

Abstract: Through the advancement in technology and the spreading of the internet, a wide range of different application types has evolved. To cope with the increased complexity of today's Web-based systems, approaches facilitating development, operations and evolution of these heterogeneous and distributed systems are vital. The approach, presented in this contribution, is based on a dedicated information model allowing the description of the overall system and its components characteristics. At runtime, the relevant model information is processed and provided to different in the system's lifecycle involved stakeholders and supports them to fulfill their activities. Due to this use in terms of orientation and guidance, our approach is called the Integrated Information Map (i²map). Furthermore, the descriptive information regarding the nominal status and behavior of the components is used for automated monitoring and testing to assure the quality of the overall system at deployment and throughout operations.

1 INTRODUCTION

Through the advancement in technology and the spreading of the internet, the World Wide Web has evolved from a decentralized information medium to a platform for a wide range of different application types (Phifer, Kenney, Genovese et al., 2006). These Web-based systems support collaboration scenarios, deliver complex services and are often characterized by the integration and composition of functionalities and content from different organizations and software systems.

In this context the question regarding the development, operation and strategic and technical evolution of such highly distributed and heterogeneous Web-based systems arises. As in most cases, the applications are rather developed iterative than from scratch and make tremendous use of existing components, the challenge of finding and binding of heterogeneous building blocks has to be addressed, making detailed descriptions about the capabilities and configuration of system components fundamental. Depending on the state of the application regarding its lifecycle, such relevant information must be provided to different stakeholders to support their activities in an adequate way. Furthermore, the availability of the overall system at a high level of quality has to be assured. On the one hand, this motivates the need of mechanisms to preview or detect malfunctions of all kinds of components, which in such an extremely interdependent system can have effects on the overall system and provide support for remedial actions. On the other hand, the building blocks and the system as a whole have to be observed regarding the quality they offer their services, to guarantee an efficient and satisfying support of the user's need.

In this paper we introduce an information model as the base for a dedicated approach for the lifecycle support of highly distributed Web-based systems. In section 2, we describe a real-world scenario from a large-scale EAI-project at the University of Karlsruhe and define functional requirements regarding the problem scope. In section 3, we present excerpts of the underlying information model, providing information about the system composition and components characteristics. Section 4 defines architectural concepts for the runtime use of the descriptive information to realize the defined functional requirements and the implementation of a support system. In section 5, we give a brief overview of related approaches and conclude with a summary and an outline of planned future work in section 6.

2 THE PROBLEM DOMAIN

In the following section, we introduce fundamental functional requirements for supporting development, operations and evolution of highly heterogeneous and distributed systems. The requirement analysis was based on the experience gained in several large-scale industry EAI projects.

2.1 An Example SOA Scenario

As an example scenario, we present the project "Karlsruhe's Integrated Information Management (KIM)" (Juling, 2005) in which we have been collaborating in for several years. Within the first three-year project phase, one of our main goals was to develop a Web portal for all students of the university. The portal shall serve as a uniform access point to all study-relevant information and business processes and provide novel features realized by integrating diverse legacy systems and processes.

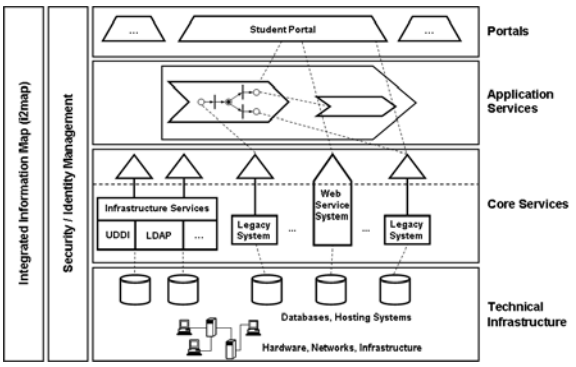


Figure 1: Architecture of the KIM project

In order to cope with the integration challenges described above, the KIM project is founded on a Service-oriented Architecture (SOA) as depicted in Figure 1. The Core Services layer mainly comprises highly reusable Web service wrappers with a standardized CRUDS interface (IBM Corporation, 2006). Each service provides access to a semantically cohesive set of business objects stored in a legacy system or database. For example, we have Core Services for courses, persons and rooms. An Application Service is also a Web service which composes Core Services to realize value-added functions. For example, by orchestrating the Core Services for exam results, persons and events, we provide an Application Service for creating a 'Transcript of Records' which is a detailed overview of a student's examination performances. The Portal layer comprises mainly Web portals, e.g. the student portal, which provides a central user interface for accessing the highly distributed Application and Core Services. Furthermore, the orthogonal Security and Identity Management layer achieves federated authentication, authorization and identity management concepts using a WS-Federation-based Identity Federation System (Meinecke, Nussbaumer, and Gaedke, 2005). Finally, the orthogonal Integrated Information Map (i²map) layer comprises concepts and thereon developed Web applications for describing, managing and monitoring the highly distributed system landscape.

The introduced project, with the portal for all students of the university as a central application, represents a typical example of a highly distributed Web-based system. Complexity arises on the one hand from the high amount and diversity of users (approx. 30.000 students) and participating organizational units (e.g. faculties, administration, library, other universities). On the other hand, through the portal integration of business processes spanning over different organizational boundaries and data from heterogeneous information systems. All together, the resulting system architecture can hardly be overlooked. With the necessity of delivering the services in an efficient and problem-free way, a support system is vital.

2.2 Functional Requirements

In cooperation with different stakeholders groups we have developed a set of functional requirements supporting development, operation and evolution of highly distributed systems. Each stakeholder group can be characterized by different information needs. The group of end-users includes students as well as administrative staff who use the different information systems of the university. Through the development of new (partly under reuse) and the advancement of existing functionalities, the group of developers make the largest contribution to the development and evolution of the system landscape, whereas the operators deal with the operation of the

Majer, F., Nussbaumer M., Gaedke M.: A Descriptive Approach for the Lifecycle Support of Distributed Web-based Systems. In Proceedings of 4th International Conference on Web Information Systems and Technologies (WEBIST-2008), 4-7 May, 2008, Funchal, Portugal.

resulting system. This includes tasks like the monitoring of the technical infrastructure, the legacy systems and all developed iSOA components and interfaces, and supporting the end-users dealing with diverse information systems.

The information needs and resulting functional requirements can be divided into two domains: *description* (R1-R7) and *monitoring* (R8–R11).

R1 – System overview: For the (strategic) evolution and operation of the highly distributed system, dedicated views of the system landscape are essential. This includes particularly the visualization of relations between different components.

R2 – Detailed component description: Besides general, organizational and functional descriptions, the different stakeholders need explicit information regarding aspects like security and quality of each system component.

R3 – Reuse support: To foster the overall reuse of software components and artifacts, dedicated search capabilities with the possibility to access and use desired resources are essential.

R4 – Change log: To cope with the high dynamics of a distributed system, a central log should protocol and publish all the changes made to any component of the system.

R5 – Snapshot: Information describing the current system composition and configuration, as well as to distinctive former states must be provided, to support fault analysis and increase in efficiency.

R6 – Simulation: The impact of changes through modification of a particular component, as well as adding or removing any components to the overall system, should be determined by simulation based on the system description.

R7 – Operational concept: Access to the operational concept descriptions should be provided at a central point.

R8 – System status / Error detection: Besides the possibility of requesting a component's nominal status (R2), i.e. service level agreements, techniques for the comparisons regarding their actual values must be available. Furthermore this status information should be evaluated and documented.

R9 – Automatic Testing / Auditing: For the initial deployment and during operations of system components, dedicated methodologies for the automated testing regarding functional requirements and guidelines (e.g. accessibility (W3C, 2006), security aspects etc.) should be provided.

R10 – Reporting: The type and quantity of a component's usage should be documented continuously and used while planning technological and strategic evolution.

R11 – Notification: Through messaging mechanisms dedicated persons or (external) systems should be notified about incidents (i.e. system disturbance through the malfunction of a single component) in the distributed system.

To support the various stakeholders appropriately regarding the lifecycle support of highly distributed systems, the different features should be integrated and combined with each other in a support system. Therefore, each functional requirement must be implemented in a highly adaptable way to cope with the needs and interests of each user group and provide dedicated views on the relevant information.

3 AN INFORMATION MODEL FOR HIGHLY DISTRIBUTED WEB-BASED SYSTEMS

The analysis of the identified functional requirements of the last section leads to the conclusion that, regardless of the concrete realization of any of the functionalities, detailed information about each component, as well as the relations among each other, are fundamental. Moreover, beside descriptive aspects, e.g. to foster development and evolution through reuse or composition of existing components, in particular information about the expected behavior of all components is essential for a smooth operation of the overall system.

(Ackermann, Brinkop, Conrad et al., 2002) define the dimensions *marketing*, *task*, *terminology*, *quality*, *coordination*, *behavior* and *interface* for the description of business components and suggest dedicated and well-known standards for the notation to represent the information and to simplify reusability between organizations. In regard to highly distributed systems, we have identified a comprehensive set of conceivable building blocks of system architectures and specified their relevant characteristics in the aforementioned dimensions. The resulting information model forms the base for a broad specification and through this the monitoring of the system.

3.1 Modeling Distributed Systems

The model comprises a metadata concept that is based on the Dublin Core Metadata Initiative (Andresen, 2003), a de-facto standard defining a set of common meta-level attributes (*ContentObject*).

By building upon this standard and introducing further attribute definitions, the central interface *iSOAComponent* (integrated service-oriented architecture component) captures characteristics all system building blocks have in common. This includes components of the technical infrastructure like servers, data bases and legacy systems and components more specific to highly distributed and Web-based systems (*SOAComponent*). The latter contain building blocks and therewith related type definitions for Web applications (e.g. Application, Domain and Audience), an abstract class for Web services and classes for specific building blocks, e.g. SecurityRealm for modeling organizational zones of control over networks, hardware and software systems. The type WebService is separated into application services, core services and infrastructure services. The latter serve for the modeling of fundamental infrastructure Web services, like the identity provider or security token service used for federated identity management scenarios.

Figure 2 shows an excerpt of the UML-based representation of the information model with the definition of iSOAComponent and several associated types. As depicted, each iSOAComponent includes metadata conform to the Dublin Core standard and further elementary information such as the component's architectural layer affiliation (*Layer*). However the Status does not imply the actual operational status, but allows the assignment of a component to different phases of its lifecycle (e.g. implemented, tested and operational). Accordingly there is the possibility to specify contact information regarding different concerns as well as information about the component's changes and versions (*ChangeLog*), and the technical documentation (*Documentation*).

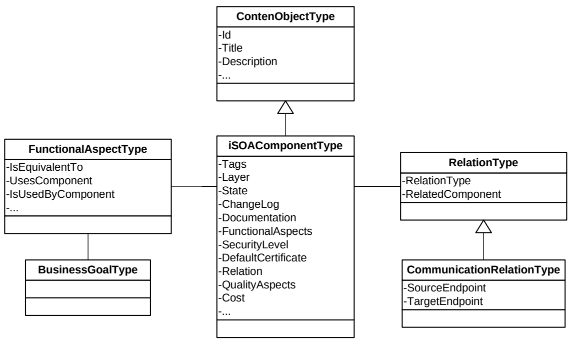


Figure 2: Meta-information concept

General security aspects like the classification of a component according to predefined security levels (*SecurityLevel*), or the default Certificate regarding communication (*DefaultCertificate*) are integrated directly into iSOAComponent. More complex and specific security concerns as authorization policies for operation invocations or transport and message security aspects, as well as the interface description, are realized by the definition of endpoints. Thereby the degree of freedom of several endpoints for one component, allows the specification of multiple types of access with different protocols and policies (i.e. certificates).

The description of the functional aspects of a component can be realized with the type *FunctionalAspectType*. As depicted in Figure 2, it enables the description of a component's general functionality and field of application, as well as the business goal it supports. With the declaration of attributes as *IsEquivalentTo*, *UsesComponent* and *IsUsedByComponent*, functional relations and dependencies between components can be modeled. E.g., this permits one to gain an overview on the effect the breakdown of one component has on the overall system. Furthermore, the attribute *Relation* facilitates the declaration of further relations.

The definition of *QualityAspects* enables the specification and validation of quality parameters for components. Besides service level agreements, this includes the ability to define functional tests.

3.2 Modeling Web-based Systems

To model specific architectural aspects of Web applications, another excerpt of the information model is shown in Figure 3.

As described in 3.1, type definitions for all components comprise a general set of attributes from iSOAComponent (cf. figure 2). Specific components of Web-based systems (e.g. *PortalApplication*, *Domain*, *ControlFunction* etc.) inherit further characteristics of *SOAComponent* (cf. figure 3). For example, *ApplicableGuideline* allows the specification regarding which standards or guidelines conformity of a component should be assured at initial deployment and during operation. In this context, mechanisms as described in (Luque Centeno, Delgado Kloos, Gaedke et al., 2006) can be utilized to guarantee the accessibility of content.

The WebComposition approach (Gaedke and Turowski, 2002) describes the development of Web applications through an open process model, allowing the integration of arbitrary processes, with reuse of existing components as an integral element. The information model follows this concept as the modeling of a Web application is based on the *PortalApplicationType* and *ClientApplicationType* respectively, which comprises several application domains. Each domain has the ability to include further domains and provides a desired functionality as part of a Web page. Depending on the expected audience and correlated configuration, the domain utilizes a dedicated building block, the control function, to realize the functionality. Hence, beside the execution of the functionality, certain kind of control functions are responsible for requesting and processing data from Web services or other data providers which is modeled by specifying relations from the particular control function to the corresponding system component.

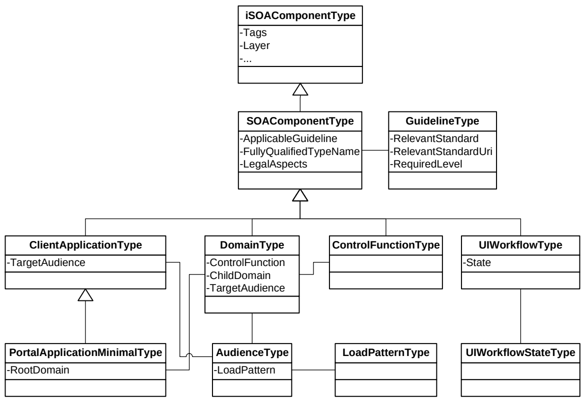


Figure 3: Excerpt of Web-based system aspects

The consideration of the growing integration of Web application in the execution of business processes, motivates the need for modeling “the computerized facilitation or automation of a business process, in whole or part” (Hollingsworth, 1995), the so-called user interaction workflow, as an imminent aspect. With the definition of *UIWorkflows* and the correlation to diverse domains and control functions, appropriate interaction structures for different audiences and the management of the communication with underlying IT systems can be modeled for each task. During operation of a complex Web-based system, this modeling information allows an efficiency analysis of the defined user interaction workflows as well as the valuation of the system disturbance through the malfunction of a single component.

4 THE INFORMATION MODEL APPLIED

To provide appropriate stakeholder support, the functional requirements described in section 2 have to be realized highly adaptable. Moreover, the integration of these functionalities into a dedicated support system as well as existing portals or client applications is essential.

As some functional requirements emphasize the description and others the monitoring of the components and the overall architecture, in the following basic concepts and building blocks for their realization are described separately. Both approaches are characterized by the strong incorporation of model information at runtime. Furthermore, they are consistently oriented towards the presented architecture of the KIM project as they form an imminent part of it but can be applied to any distributed Web-based system environment.

At the end of this section, we present an overview of the actual realized functions integrated in a Web application.

4.1 Support by Description

The realization of descriptive functionalities regarding the components (*Managed Objects*) and the overall architecture of the highly distributed system is based on the registration of all components at a central registry. Thereby, the relevant information is specified according to the type of the component and in compliance to the information model. Depending on the management ability of the registered component, this step can be conducted automatically.

As the registry constitutes the central element responsible for the model information, it provides access to the data via several interfaces. Besides an interface for the retrieval of the information according to schemas based on the information model (*Registry*), the registry publishes subsets of the information conforming with the UDDI standard (Bellwood, Clément, Ehnebuske et al., 2004) or for other models like e.g. the WebComposition Architecture Model (Meinecke, Gaedke, Majer et al., 2006) and therefore supports the application of the information in different contexts (cf. Figure 4). For scenarios with participating autarkic organizational units but inter-organizational business processes or other system correlations, the federation of such registries is applicable. This assures the characteristic of loosely coupled system interaction specific to the Web and Service-oriented Architectures, but allows the support of description and monitoring transcending organizational boundaries.

The descriptive functional requirements, e.g. system overview (R1), detailed component description (R2) and reuse support (R3) are actually realized by a dedicated application service (*Description*). Depending on the chosen portal functionality, this component queries the registry for the relevant information and prepares it according to the user needs. Furthermore, it provides access to documents related to components (e.g. conceptual models, technical documentation and source code) possibly stored in different Repositories.

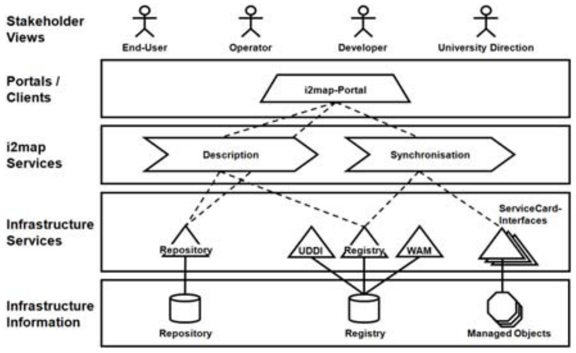


Figure 4: Architectural concept supporting description

Another application service (*Synchronization*) guarantees consistency between the stored data in the registry and information, components with management ability publish about themselves. On the one hand, it propagates changes carried out to a component itself (e.g. changes to the interface) to the registry. On the other hand, it passes modifications to registry entries regarding available information at components (e.g. updates about responsibilities of a components made via a support system) to the component for further processing.

4.2 Support by Monitoring

Besides descriptive information about the system characteristics and behavior, the actual performance of each component as well as the overall system is essential to conduct a variance comparison and to detect inefficiency and failure during operation. The achievement of the functional requirements in the monitoring dimension depends to a great extent on the management ability of the system components.

The application service *Monitoring and Event Processing* coordinates the monitoring of the different components. According to its configuration, it accesses the registry to query the identifiers and end points of components to monitor and passes the information to *Observers* responsible for the actual surveillance (cf. Figure 5). The observers are realized as CRUDS core services and provide dedicated monitoring metrics for different component types in a highly distributed and heterogeneous system. On the one hand, the indirection between the application service and the managed objects via observers has the advantage of easy scalability and the parallelization of the status requests for the overall system. On the other hand, the encapsulation of the concrete logic for the monitoring of a certain type of component regarding a specific dimension increases the flexibility and coverage of the approach. For example, this enables the use of several observers differing in monitoring aspects and metrics for a specific component type or the application of observers consuming the data of existing monitoring systems (e.g. in the area of the technical infrastructure).

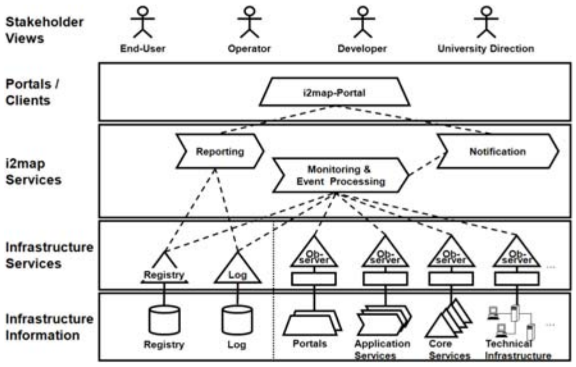


Figure 5: Architectural concept supporting monitoring

The results of the continuous and automated status querying and testing, conducted by the observers, are processed by the application service (monitoring and event processing) according to predefined rules and stored into a *Log*. Identified and anticipated system disturbances through malfunctions of components are reported via the application service *Notification* to responsible persons or (external) systems. Furthermore, the *Reporting* service provides access to the stored runtime information and prepares it depending on the stakeholder's needs, e.g. the average availability and other quality aspects or the usage of a component for billing purposes.

4.3 i^{2}map Tool Support

In the following, we present the application of the described information model and the architectural concepts in the KIM project at the University of Karlsruhe. By the use of the model information at runtime, the developed support system provides guidance and orientation for different stakeholder according to a "map".

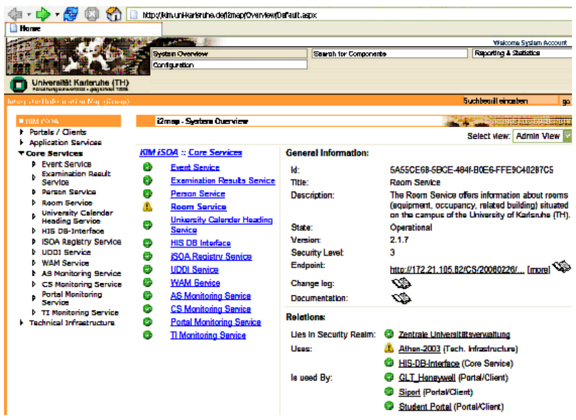
To provide the necessary information about the components, we implemented the designated management operations *getServiceCard* and *getStatus* of each existing core and application service,

Majer, F., Nussbaumer M., Gaedke M.: A Descriptive Approach for the Lifecycle Support of Distributed Web-based Systems. In Proceedings of 4th International Conference on Web Information Systems and Technologies (WEBIST-2008), 4-7 May, 2008, Funchal, Portugal.

allowing the retrieval of nominal (ServiceCard) and actual (Status) data. In a further step, all relevant components were registered at a central KIM-Registry and data regarding components without management abilities was gathered manually.

For the monitoring of the core and application services, observers were implemented, examining managed objects according to the Web Service Level Agreements project (Keller, Kar, Ludwig et al., 2002). Concerning the monitoring of the technical infrastructure, as well as further components, e.g. the Microsoft BizTalk Server 2006, dedicated observers use existing monitoring systems like Nagios (Galstad, 2006) or Microsoft System Center Operations Manager 2007 (Microsoft, 2006) of the university's computer center.

Finally the relevant information is made accessible via dedicated application services in the i²map portal, based on the Microsoft Office SharePoint Server 2007 (cf. Figure 6). In addition to administrative support (e.g. for the registration of new components), the developers use the actual implementation particularly for searching for specific components, as well as to access the detailed descriptions. Besides the operators, who are mainly supported by first views on the system status, the university direction gets an overview regarding the utilization of specific services. Furthermore, for the group of end users the portal provides a view on upcoming system maintenance and a listing of current system breakdowns.



KIM iSOA :: Core Services

- Event Service
- Examination Results Service
- Person Service
- Room Service
- University Calendar Heading Service
- HIS DB Interface
- iSOA Registry Service
- UDDI Service
- WAM Service
- AS Monitoring Service
- CS Monitoring Service
- Portal Monitoring Service
- TI Monitoring Service

General Information:

Id: 5A55CE68-5BCE-484f-B0E6-FFE9C40287C5
Title: Room Service
Description: The Room Service offers information about rooms (equipment, occupancy, related building) situated on the campus of the University of Karlsruhe (TH).
State: Operational
Version: 2.1.7
Security Level: 3
Endpoint: http://172.21.105.82/CS/20060226/... [more]
Change log:
Documentation:

Relations:

Ues in Security Realm: Zentrale Universitätsverwaltung
Uses: Athen-2003 (Tech. Infrastructure)
HIS-DB-Interface (Core Service)
Is used By: GLT_Honeywell (Portal/Client)
Siport (Portal/Client)
Student Portal (Portal/Client)

Figure 6: Overall system's status in the i²map Portal

5 RELATED WORK

In the following, several existing approaches facing the challenge of supporting development, operation and evolution of highly distributed Web-based systems will be described briefly.

When analyzing the state of the art, an important group to be considered, are approaches from the Web Engineering research community (e.g. WEBML (Ceri, Fraternali and Bongio 2000), OOHDM (Schwabe, Rossi and Barbosa, 1996)) dealing with the systematic construction of Web applications with particular consideration of Web-specific characteristics. As these methodologies emphasize the modeling and development of Web applications built from scratch, they focus less on operation and evolution of highly distributed Web application. Regarding the consideration of existing components of Web-based systems, the WebComposition approach based on the principles of evolution and Component-Based Web Engineering (CBWE) is an exception. It naturally includes compositional, integrative and federated aspects and some of the WebComposition approach's core principles and concepts could be adopted to our solution.

For the modeling of heterogeneous and distributed Web-based systems, there exist several concepts for domain-specific aspects, e.g. regarding the specification of the quality performance of Web services (OASIS, 2005). Moreover, (OASIS, 2006b) deals with complex systems and defines elementary building blocks and concepts for Service-oriented Architectures in an abstract manner to achieve a common understanding and a clear terminology independently of actual implementations and technologies. Other approaches like (Kirchner, 2005) and (Winter, Brigl and Wendt, 2003) allocate building blocks of distributed systems to architectural layers and use descriptive information to realize first descriptive functionalities. Even though, the information is utilized at runtime, the approaches focus on constricted scenarios and often neglect federated aspects of today's distributed Web-based systems.

Regarding the operation of systems, (OASIS, 2006a) provides a Web service-based framework for the management of components, especially for Web services. Furthermore there exist solutions (partly from the industry) for the monitoring of portals, as well as the technical infrastructure (e.g. ManageEngine; Nagios) but these approaches mostly concentrate on specific aspects. However Microsoft's Dynamic System Initiative aims at a comprehensive approach for design, installation and operations of distributed systems (Turner, 2006). The methodology is based on the System Definition Model, which will be replaced by the Service Modeling Language in future, facilitating the modeling of abstract service components.

Microsoft's vision is to use this information for the configuration of components at installation or for the monitoring of those services with the System Center Operations Manager 2007.

6 CONCLUSION

To cope with the increased complexity of today's Web-based systems, approaches facilitating development, operations and evolution of these heterogeneous and distributed systems are vital. In this context the paper presented functional requirements for a support system, focusing on the aspects of describing and monitoring the overall system. To realize these functionalities, we presented an information model applicable for modeling the relevant aspects and characteristics of highly distributed Web-based systems. Together with the architectural concepts using the modeled information at runtime, the resulting approach supports different stakeholders during their activities within an application's lifecycle. The realization of first functionalities and their integration within the Integrated Information Map of the KIM project at the University of Karlsruhe (TH), showed the support potential of the approach in a highly distributed, heterogeneous and Web-based system environment.

In the future, we will work on further implementations of descriptive functionalities like the snapshot (R5), as well as the simulation of changes to distributed Web applications (R6). Moreover, we will focus on further realization of monitoring aspects as the pilot operation phase of the student's Web portal and all correlated components is scheduled for the last quarter of 2007. Particularly, this includes the development of observers with dedicated monitoring and test metrics (R9) to assure the quality for specific component types. Finally the integration of further functionalities for the remote management of components into the i²map portal and the development of concepts regarding self-healing and reconfiguration of distributed systems are of interest.

As the presented approach is the result of more than two years research, not all aspects could be described in detail. Thus, we are working on further publications focusing e.g. on the use of the descriptive information at runtime and the integration of model information of existing (Web Engineering) approaches into our approach. A transformation of the existing data according to our information model would allow supporting operations and evolution of Web-based systems developed with these methodologies.

REFERENCES

Ackermann, J., Brinkop, F., Conrad, S., et al., 2002. *Standardized Specification of Business Components*.

Andresen, L., 2003. *Dublin Core Metadata Element Set, Version 1.1: Reference Description*, DCMI.

Bellwood, T., Clément, L., Ehnebuske, D., et al., 2004. *UDDI Version 3.0*, UDDI.org.

Ceri, S., Fraternali, P., and Bongio, A., 2000. *Web Modeling Language (WebML): A Modeling Language for Designing Web Sites* In 9th International World Wide Web Conference (WWW), 2000, 137-157.

Gaedke, M., Turowski, K., 2002: *Specification of Components Based on the WebComposition Component Model*.

Galstad, E., 2006. Nagios Homepage.

Hollingsworth, D., 1995. *The Workflow Reference Model (1.1)*. The Workflow Management Coalition.

IBM Corporation, 2006. *Elements of Service-Oriented Analysis and Design.* IBM Homepage.

Juling, W., 2005. *KIM Homepage*. University of Karlsruhe

Keller, A., Kar, G., Ludwig, H., et al., 2002. *Managing Dynamic Services: A Contract Based Approach to a Conceptual Architecture*. EMISA´05. Austria, 2005.

Kirchner, L., 2005. *Cost Oriented Modelling of IT-Landscapes: Generic Language Concepts of a Domain Specific Language.*

Luque Centeno, V., Delgado Kloos, C., Gaedke, M., et al., 2006. *Web Composition with WCAG in mind. In Journal of Web Engineering (JWE).*Rinton Press. 2006

Microsoft Corporation, 2006. Microsoft System Center Homepage.

Meinecke, J., Gaedke, M., Majer, F., et al., 2006. *Capturing the Essentials of Federated Systems*. In 15th International World Wide Web Conference.

Meinecke, J., Nussbaumer, M., Gaedke, M., 2005. *Building Blocks for Identity Federations*. ICWE 2005, Sydney, Australia, 2005.

OASIS, 2005. *Quality Model for Web Services*.

OASIS, 2006a. *Web Services Distributed Management*.

OASIS, 2006b. *Reference Model for Service Oriented Architecture 1.0*.

Phifer, G., Kenney, L.F., Genovese, Y., et al., 2006. *Hype Cycle for Web Technologies, Research Report*. Gartner Research. Stanford, CT.

Schwabe, D., Rossi, G., and Barbosa, S., 1996. *Systematic Hypermedia Design with OOHDM*. In ACM International Conference on Hypertext. USA, 1996.

Turner, M., 2006. *Microsoft System Center takes on enterprise IT management market leaders.*

Winter, A., Brigl, B., Wendt, T., 2003. *Modeling hospital information systems. The revised three-layer graph-based meta model 3LGM*. Stuttgart, Germany.

W3C, 2006. Web Accessibility Initiative (WAI) Homepage (2006): http://www.w3.org/WAI/