Patrick Freudenstein, Marko Boettger, and Martin Nussbaumer: Efficacious Reuse Support as Enabler for Cross-Methodological Web Engineering with Stakeholders. In Proceedings of 8th International Conference on Web Engineering (ICWE2008), 14-18 July 2008, New York, USA.

Efficacious Reuse Support as Enabler for Cross-Methodological Web Engineering with Stakeholders

Patrick Freudenstein[1], Marko Boettger[2] and Martin Nussbaumer[1]
Karlsruhe Institute of Technology - University of Karlsruhe (TH)
Department of Telematics, IT Management and Web Engineering Research Group,
Engesserstr. 4, 76128 Karlsruhe, Germany
[1]{patrick.freudenstein, martin.nussbaumer}@kit.edu, [2]s_boettg@ira.uka.de

Abstract

As well as in traditional software engineering, reuse is a key factor for the efficient and efficacious construction and evolution of complex Web applications. Efficaciously integrated in the development process and supported by an appropriate framework, reuse significantly increases development speed, application quality and, by enabling rapid prototyping, improves stakeholder communications. Although reuse aspects in traditional software engineering have been thoroughly investigated, their successful adoption to the Web Engineering domain still remains nontrivial leaving room for dedicated Web-specific solutions. Beyond that, current consolidation activities in the Web Engineering community underline the significance of a unifying solution. To this end, we present a comprehensive ontology-based Web Engineering reuse approach that establishes a common basis for cross-methodology reuse strategies and emphasizes stakeholder collaboration. To unfold the full potential of reuse, it incorporates both planned and spontaneous reuse strategies. We point out the approach's extensibility and applicability for cross-methodological Web Engineering and demonstrate how it considerably accelerates development speed and improves stakeholder communications.

1. Introduction

Reuse has been identified very early as an important software engineering principle being able to significantly improve development efficiency and quality [21]. In fact, reuse can lead to greater schedule and effort savings than any other rapid-development practice – if implemented as a systematic and dedicated long-term strategy and supported by an efficacious framework [20].

In the Web Engineering research field, aspects of reuse have primarily been examined in the context of a particular Web Engineering method and focusing on specific artifact types like models or components, e.g. OOHDM [27], WebComposition [14] or WebML [6]. While most of the Web Engineering approaches describe their modeling methodology's adequacy for reuse, the efficient and efficacious realization of reuse when developing Web applications still remains nontrivial.

Beyond that, consolidation efforts like the Model-driven Web Engineering initiative *MDWEnet* [32] or research papers, e.g. [28], strive for achieving interoperability between common Web Engineering methodologies, e.g. OO-H [16], UWE [18] or WebML [5]. Thereby, not only the significance of a unifying reuse approach is emphasized, but also the immense potential of reuse in interoperable, *cross-methodological* Web Engineering scenarios is underlined.

One aspect, which has not been considered in reuse-related Web Engineering research yet, is the integration of stakeholders and their specific characteristics. Several studies, e.g. [29], have proved the strong correlation of a project's success and intense stakeholder involvement in all phases of the development process. Consequently, enabling stakeholders to participate in reuse processes is a key factor. In the most cases, adapting existing artifacts is much easier than creating new artifacts from scratch - especially for people with few technical skills. Thus, empowering stakeholders to find reusable artifacts, methodologies and tools suitable both for the given problem and their individual knowledge and skills, seems to be promising.

Patrick Freudenstein, Marko Boettger, and Martin Nussbaumer: Efficacious Reuse Support as Enabler for Cross-Methodological Web Engineering with Stakeholders. In Proceedings of 8th International Conference on Web Engineering (ICWE2008), 14-18 July 2008, New York, USA.

To this end, we present a cross-methodological Web Engineering reuse approach focusing efficiency and efficacy of reuse-related tasks. A generic reuse ontology for the Web Engineering domain containing concepts and relations for methodologies, artifact types, processes, tasks, concerns, application types, stakeholder types, knowledge etc. serves for the efficacious finding and registering of artifacts. The presented approach is not restricted to particular artifact types or Web Engineering methodologies; it rather provides an open and integrative framework with dedicated extension points. In order to unfold the full potential of reuse and reflecting the reuse scenarios faced in the collaborative and distributed development of large-scale Web applications, we present coordinative solutions both for planned *and* spontaneous reuse. Supplementing this conceptual approach, we present an associated federative architectural framework for distributed repositories and clients which was designed as open and non-invasive as possible.

In section 2, we illustrate reuse-related challenges and potentials in the context of the construction of large-scale Web applications. A core requirements catalogue for a systematic Web Engineering reuse approach serving as starting point for our research and for evaluating existing approaches concludes the chapter. An overview of the approach's core ideas and concepts is given in section 3, whereas a detailed description including how to apply our approach to existing Web Engineering methodologies and thus realize the potential of cross-methodological reuse follows in section 4. The approach's applicability and benefits, e.g. considerably accelerated development speed and improved stakeholder communications, are demonstrated in section 5 based on an example scenario. Section 6 evaluates the state of the art and presents related activities underlining the significance of the presented approach. Finally, section 7 concludes the paper and outlines future work.

2. Problem Scope

In this section, we first illustrate several representative reuse-related challenges faced in the construction of large-scale Web applications in real-world projects. Afterwards, a core set of requirements a systematic Web Engineering reuse approach should fulfill is presented. These requirements served as a starting point for the design of our approach and were derived both from our experiences in many real-world projects as well as from challenges and requirements found in literature.

2.1 Web Engineering Reuse Challenges

A significant characteristic for projects dealing with the development of large-scale enterprise Web applications is the multitude and diversity of the involved stakeholders. In order to assure the project's success by efficient communication and collaboration throughout all phases [29], it is essential to employ dedicated modeling techniques, templates and tools appropriate for the individual characteristics of stakeholders (e.g. knowledge, professional background etc.). This insight lead to the DSL-based Web Engineering approach [22, 23] which aims at providing dedicated languages for the diverse aspects of Web applications (workflows, dialogs, sitemap etc.), each of them offering various modeling notations tailored to a specific stakeholder group. Nonetheless, any Web Engineering methodology could provide stakeholder-specific notations and define associated model transformations to their current modeling techniques. In a larger *cross-methodological* context, it is imaginable that the choice of the Web Engineering methodology used for the realization of a particular feature is made depending on the given stakeholders' skills and the qualifications required by the methodologies.

Thus, when searching for or storing reusable artifacts, *stakeholder characteristics* should be taken into account – irrespective from the Web Engineering methodology used. Beyond that, in order to enable stakeholders to contribute to the development effort by adapting existing models or even creating new ones based on templates, a reuse strategy and its associated framework should include stakeholders and domain experts having only little technical skills in their list of target audiences.

As the positive effects of reuse are not restricted to particular types of artifacts, a systematic Web Engineering reuse approach should be *generic* in terms of supporting any type of artifact occurring in the development process [11]. Thereby, it is desirable to *non-invasively* build on existing artifact stores, e.g. document repositories, model databases, component repositories or version control systems. Beyond that, a reuse strategy should be *independent* from the development methodology used. This means, that an adequate reuse approach should provide positive impact on any Web Engineering methodology and should establish a common basis for *cross-methodological reuse*. Especially in the context of the above-mentioned consolidation efforts like *MDWENet* striving for interoperability between today's established Web Engineering methodologies and their

Patrick Freudenstein, Marko Boettger, and Martin Nussbaumer: Efficacious Reuse Support as Enabler for Cross-Methodological Web Engineering with Stakeholders. In Proceedings of 8th International Conference on Web Engineering (ICWE2008), 14-18 July 2008, New York, USA.

tools, a unifying approach unfolding the power of cross-methodological reuse is desirable.

When developing with reuse, efficiently and efficaciously *finding suitable reuse assets* is crucial [19]. Thereby, searching on a keyword or full-text basis is usually not sufficient. In fact, an appropriate search mechanism should strongly incorporate the actual *context* [30], e.g. the project and application type, the given task and process phase, the involved stakeholders, the feature's associated business domain, the Web-specific concern etc. Such complex context-dependent search queries are often not directly resolvable, but rather require knowledge-based resolution strategies. Thus, powerful semantic inference-enabled search capabilities tailored to the Web Engineering domain should be provided. Especially for users having little experience in searching for suitable artifacts, it can be difficult to determine good search parameters. To this end, enabling users to browse through the registry space can further increase efficiency and efficacy [9].

Having found a suitable artifact, *reusing and integrating* it should be very efficiently. Therefore, finding and retrieving artifacts should be possible within the specific proprietary tools and applications where they are used in. For example, business process models and templates should be directly searchable and retrievable from within the associated business process modeling tool. In the context of reusing software components, it is desirable to have direct installation and integration capabilities at runtime, ideally augmented by *safe preview* facilities for integration testing [26].

Besides assisting in development with reuse, a systematic reuse approach should also support reuse-related tasks in the context of *development for reuse*. In order to achieve an active participation in the reuse strategy, *registering and storing artifacts* for reuse should require as little effort as possible but still be efficacious in terms of the provided metadata. Therefore, methodologies for the automated derivation of comprehensive metadata from the actual context should be considered [3].

In large projects or organizations, it happens quite often that a particular artifact is needed by several parties but does not exist in the repository. Then, each party individually starts with developing for reuse, which in turn leads to a considerable amount of redundant development effort. Supported by an efficacious *coordination mechanism* indicating ongoing development efforts very early, such parallel developments could be efficiently aligned [33].

2.2 Requirements Catalogue

Strong stakeholder orientation: Stakeholder characteristics should be treated as an important context parameter for storing and finding artifacts. Moreover, in order to enable stakeholders to participate in the reuse strategy, efficaciously storing and finding artifacts should be rather intuitive, requiring only little technical knowledge.

Generality: All types of artifacts along with their type-specific stores should be non-invasively integrated. Moreover, the approach should establish a common basis for cross-methodological reuse. Therefore, its applicability to today's Web Engineering methodologies, tools and frameworks should be assured.

Efficient and efficacious search: Powerful semantic context-dependent search capabilities tailored to the Web Engineering domain should be provided. In addition, facilities for browsing through the registry space should be offered.

Efficient reuse: Reusing existing artifacts should be very efficiently from within the specific tools and applications where they are used in. Regarding the reuse of software components, direct integration capabilities at runtime, ideally augmented by safe preview facilities for integration testing, are desirable.

Efficient and efficacious storing: Aiming at an active participation in the reuse strategy, registering and storing artifacts should strive for requiring as little effort as possible - without losing efficacy.

Coordination of development for reuse: A systematic reuse approach should provide coordinative support reducing redundant efforts in development for reuse.

3. The Web Engineering Reuse Sphere

Facing the presented challenges and requirements, we present a systematic Web Engineering reuse approach which we call the "Web Engineering Reuse Sphere". It is based on the idea of several spheres of distributed, ad-hoc- and infrastructure-based repositories and a semantic registry in their core (cf. section 3.1). The semantic core is represented by a dedicated reuse-related ontology for the Web Engineering domain (cf. section 3.2). Section 3.3 and 3.4 show how artifacts can be efficaciously searched and registered.

Patrick Freudenstein, Marko Boettger, and Martin Nussbaumer: Efficacious Reuse Support as Enabler for Cross-Methodological Web Engineering with Stakeholders. In Proceedings of 8th International Conference on Web Engineering (ICWE2008), 14-18 July 2008, New York, USA.

3.1 The Sphere Concept

Figure 1 depicts the sphere concept. The spheres are divided into various areas representing the different types of artifacts occurring in the Web Engineering domain, e.g. documents, models, components etc. Each area contains type-specific repositories for its reusable artifacts. Thereby, the sphere defines two levels.

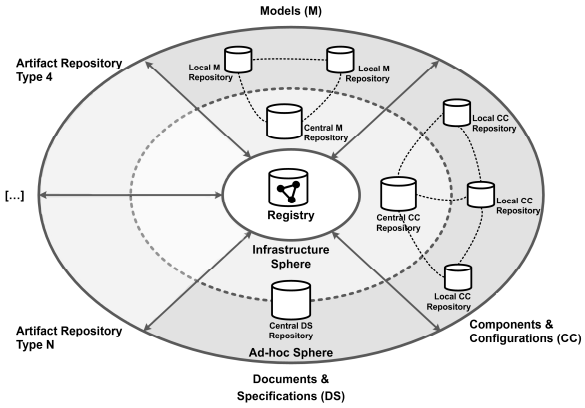

Figure 1. The Web Engineering Reuse Sphere.

The *infrastructure* level contains one dedicated reuse repository per area for planned reuse. There, sufficiently mature and stable artifacts are explicitly published for being reused. As indicated by the term 'infrastructure', such repositories are specifically set up for systematic long-term storage of artifacts including versioning.

The *ad-hoc* level is optional for an area and contains repositories for spontaneous reuse. Such ad-hoc repositories are usually already in use and are rather application-specific data stores than actual reuse repositories. In the models area, for example, a local database containing current models could be such an ad-hoc repository. Another example would be the data store of a Web application development environment running on a developer's computer and representing the current state of development. Consequently, artifacts are available in the ad-hoc level from the moment on where the user saves them for the first time until they are deleted.

A *central ontology-based registry* forms the core of the sphere. It registers all artifacts in all repositories – both on the infrastructure and the ad-hoc level – along with their semantic metadata and provides holistic registration and search functionalities. When searching for artifacts, results can encompass both artifacts that were explicitly published in a repository on the infrastructure level and artifacts from a repository on the ad-hoc-level being still under development. Reuse can thus be performed in a *peer-to-peer* style on the ad-hoc-level and in a *planned* way on the infrastructure level, whereby both mechanisms contribute to the approach's efficiency and efficacy. The former allows for discovering and exchanging work in progress between local application-specific stores. This in turn results in a *coordinated* and efficient *collaboration* by reducing redundant developments and avoiding consolidation efforts.

An interesting symptom that can be observed on the ad-hoc-level is the correlation of an artifact's popularity and its persistency. Artifacts being very popular, e.g. due to their quality, applicability, generality etc., will be more persistent than others. This is due to the fact that repository contents on the ad-hoc-level are usually only available while at least one person uses them for their current project. When a person removes an artifact from their local repository and the artifact is not contained in any other repository on the ad-hoc level, i.e. nobody else (re-)uses this artifact, it is no longer available. Analyzing factors like an artifact's degree of persistency or its (re-)usage in various settings can thus help to derive statements about its characteristics like e.g. its quality, applicability, usefulness etc.

After an artifact was completed and has gained sufficient maturity, e.g. by passing quality inspections, it can be transferred to a repository on the infrastructure level, thus being persistently and reliably available for planned reuse.

3.2 The Semantic Core: WebE Reuse Ontology

The semantic ontology-based registry in the sphere's core is in charge of registering all artifacts throughout the repository space based on semantic metadata. Therefore, we developed a generic *Web Engineering Reuse ontology* which provides the basis for classifying artifacts as well as for powerful inference-based search mechanisms. We elaborated the ontology according to established ontology development methodologies [24, 31] and put emphasis on *generality*, i.e. keeping the ontology open for any Web Engineering method and incorporating well-defined extension points. Furthermore, we strived for integrating existing ontologies where possible. For example, the FOAF ontology [4] being related to the concept *stakeholder*, the Dublin Core ontology [2] defining standardized metadata properties for core concepts like *artifact* or *project* or the OntoWeb ontology [8] covering the concepts *product* and *business domain*.

Figure 2 gives a simplified overview of the ontology's core concepts and relations. The ontology defines concepts for *artifacts* and their context

Patrick Freudenstein, Marko Boettger, and Martin Nussbaumer: Efficacious Reuse Support as Enabler for Cross-Methodological Web Engineering with Stakeholders. In Proceedings of 8th International Conference on Web Engineering (ICWE2008), 14-18 July 2008, New York, USA.

regarding the associated *Web application*, *project*, *process model, product etc*. Furthermore, with respect to the stakeholder orientation requirement, the ontology describes the interrelation of particular *task types* occurring in the development of a *Web application*, corresponding *resolution strategies* defined by *Web Engineering methodologies* and the *skills* / *knowledge* required therefore. In addition, the ontology allows for describing representative *stakeholder* groups and their *skills*.

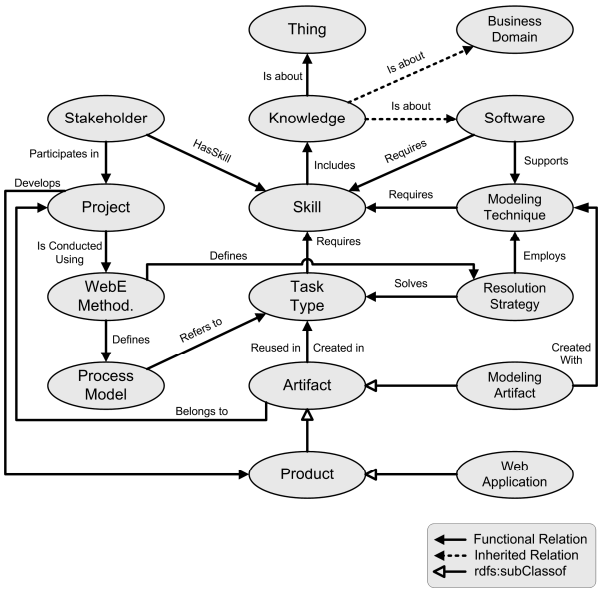

Figure 2. Simplified Overview of the Ontology

Based on the ontology, powerful knowledge-based search queries can be processed. Besides simple queries like finding existing *artifacts* being related to a particular *business domain*, *concern* (i.e. navigation, presentation etc.), *Web Engineering methodology* or *task type*, more advanced queries, especially supporting stakeholder collaboration can be resolved. In order to illustrate the following example, Figure 3 depicts a simplified excerpt from the ontology with concepts and relations from the core ontology (white ellipses), Web Engineering methodology-independent instances (grey ellipses in the middle) as well as specific instances for the Web Engineering methodologies UWE and WebML (left and right).

Thus, for the given *task type* 'design business process' and the *skills* of 'Stakeholder B' (i.e. 'BPMN modeling skills'), appropriate *modeling techniques* can be determined by inference in a first step. In this example, the query result would be the *modeling technique* 'WebML Process Modeling' which is based on BPMN and supported by the *software* 'WebRatio'. For 'Stakeholder A' having 'UML Activity Modeling' *skills*, the result would be the *modeling technique* 'UWE Process Modeling' which is based on UML and supported by the software 'ArgoUWE'.

Together with such modeling techniques, search results could directly include existing *artifacts* – in this case *modeling artifacts* - created with the same *modeling technique* in similar *project* or *Web application* type contexts. Additionally, *artifacts* representing templates for the determined *modeling technique* could be retrieved.

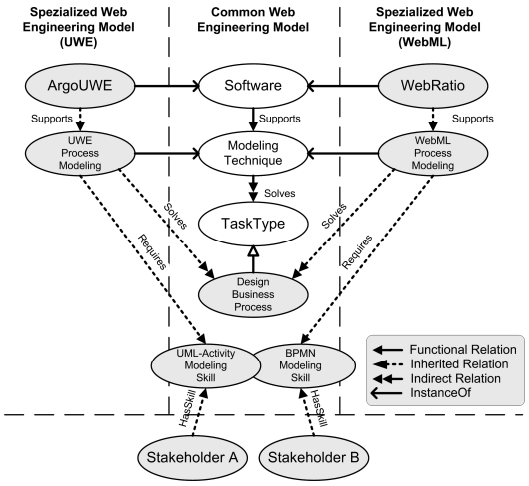

Figure 3. Ontology excerpt with instances for WebML and UWE

Such *cross-methodological* scenarios are gaining additional importance in the context of the above-mentioned consolidation activities like *MDWENet*. To this end, the Web Engineering Reuse Sphere approach and its associated architectural framework can serve as a valuable accelerator unfolding the potential of cross-methodological interchange and collaboration.

A detailed description of the ontology can be found in section 4.2. Moreover, it can be downloaded from our research homepage (cf. section 8).

3.3 Efficacious Search and Integration

In order to ease the process of finding artifacts, we strived for search mechanisms being both easy to use and efficacious in terms of finding adequate results very quickly. Common search facilities, e.g. Google, usually offer a simple mode, i.e. one input parameter for all kinds of search terms, and an advanced mode, i.e. lots of query parameters. When inexperienced people use such search facilities, it can be observed that for them the simple mode is easy to use, but leads to unsatisfying search results. A lot of knowledge about adequate search terms and query syntax is

Patrick Freudenstein, Marko Boettger, and Martin Nussbaumer: Efficacious Reuse Support as Enabler for Cross-Methodological Web Engineering with Stakeholders. In Proceedings of 8th International Conference on Web Engineering (ICWE2008), 14-18 July 2008, New York, USA.

required to achieve good results. The advanced mode offers more guidance regarding search constraints, but still requires significant knowledge about adequate search terms.

Facing these problems, we propose an *extensible, user- and scenario-based methodology* for providing search facilities. First, in strong collaboration with stakeholders – both development team members and domain experts – we identify reuse scenarios and elicit relevant search parameters. Then, based on the ontology, a corresponding SPARQL [25] query template is developed. Thereby, possibly missing relations or sometimes even concepts are determined. In such a case, the ontology is extended following a systematic ontology evolution process. Finally, a suitable search dialog for the reuse scenario is developed - again in strong collaboration with stakeholders. Thereby, usability aspects in terms of providing guidance to the user and including dynamic behavior, e.g. in form of multi-step search dialogs, are key factors. At runtime, the user input from the search dialog is inserted in the corresponding SPARQL query template which is then executed on the registry's triplet store resulting in relevant artifacts.

In addition, the user can use the search results as a starting point to browse through the registry space and perform context switches following the relations defined in the ontology. For example, for a given *artifact*, all artifacts from the same *project*, *Web application type*, *business domain* etc. or created with the same *modeling technique* or *resolution strategy* could be identified. Beyond that, also more powerful inference-based context switches are possible. For example, all *artifacts* that required similar *stakeholder skills* for their creation and that were created in the same *task type* and for the same *business domain* could be retrieved. Examples for such scenario-based search dialogs and the described browsing facilities can be found in section 5.

Having found a probably suitable artifact, it should be easily and safely integrable in the current development context and artifact-specific tools. Therefore, it is desirable to perform searches and retrieve suitable artifacts directly from within artifact-specific tools and editors. To this end, we propose an *architectural framework* (cf. section 4.1) employing concepts from the field of Enterprise Application Integration (EAI). By establishing a generic Web service layer on top of the repositories and the registry and – if required – tool-specific Web service adapters on top of the registry, proprietary tools can retrieve search results including URLs from the registry and artifacts from the repositories. For example, Microsoft applications, e.g. Word, Excel, PowerPoint or Visio, can interact with external Web services based on the Research Interface [10]. Thus, e.g. reusable artifacts in form of documents or models could be directly searched and retrieved from within Word or Visio. By providing additional Web service adapters, other tools and applications can be easily integrated. When performing searches from within a tool, some search parameters can be automatically derived from the current context, e.g. the artifact type or the software with which the artifact should be editable.

However, such existing facilities for external data source integration usually allow for one-parameter searches only. In order to offer comprehensive search dialogs exploiting the full potential of advanced knowledge-based searches, plug-in-based extensions in form of specific search dialogs can be integrated in most of today's applications. Alternatively, the proposed architectural framework contains a generic Web-based search portal for finding and retrieving artifacts.

3.4 Storing artifacts with rich metadata

While registering an artifact in a repository on the infrastructure level should require as little manually entered metadata as possible, registering artifacts on the ad-hoc-level should be performed automatically in the background based on the metadata provided within the associated application - without any additional manual input. Thus, in order to minimize the amount of manually provided metadata, approaches for extracting and mapping proprietary metadata statements to the concepts and properties defined in the ontology are required. To this end, on the *ad-hoc-repository level*, our architectural framework proposes observer agents which identify new artifacts, extract metadata statements and submit them automatically to the registry (cf. section 4.1). Thereby, new artifacts become registered automatically only a few moments after their creation or modification - without requiring modifications or extensions to the existing tools or repositories. In the future, when tool and application vendors will have adopted established ontologies which were also used in the presented ontology (e.g. the Dublin Core ontology), metadata mapping efforts will significantly be reduced. Beyond that, it would be desirable that the presented ontology is taken on in the Web Engineering research community for including methodology-specific extensions and incorporating it in their associated development frameworks and tools.

On the *infrastructure level*, artifacts are either again stored and registered from within artifact-type-specific tools and applications or submitted via a generic Reuse Web Portal. In order to allow for submitting artifacts to

Patrick Freudenstein, Marko Boettger, and Martin Nussbaumer: Efficacious Reuse Support as Enabler for Cross-Methodological Web Engineering with Stakeholders. In Proceedings of 8th International Conference on Web Engineering (ICWE2008), 14-18 July 2008, New York, USA.

infrastructure repositories from within the tools they were created or modified with, dedicated extensions for communicating with registry or repository Web services as well as dialogs for entering metadata are required. If such extensions are not feasible, the Reuse Web Portal can be used to store and register artifacts.

In each case, as much metadata as possible is extracted automatically in the same way as described above for the ad-hoc-level. However, as registering artifacts on the infrastructure level is – in contrast to the ad-hoc-level - an explicit task and metadata quality requirements are much higher, it is reasonable to have the user complement the automatically derived metadata.

In order to gain even more valuable metadata automatically, deriving semantic information from the artifact's context or a user's behavior while working with an artifact seems to be a promising approach. For example, if a particular stakeholder registers an artifact that was created using a particular modeling technique, the stakeholder's current skill set can be automatically augmented by the skills that were required for the employed modeling technique, the related business domain and the used software.

Beyond that, we examined such approaches in the context of a component-based development framework. For example, by measuring how long users worked on a component regarding a particular concern (e.g. presentation, interaction etc.), we could derive meaningful statements about the major relation of the component to a particular concern. Another example we evaluated was analyzing a component's relative location on a page and thereupon (combined with other aspects) deriving statements about its type, e.g. content component, satellite, menu, landmark, login etc. Similar ideas could be easily adopted for modeling tools or integrated development environments like e.g. WebRatio, ArgoUWE or VisualWADE.

4. Realization Details

This section describes realization details regarding the architectural framework (cf. section 4.1) and the developed ontology and its extension points for incorporating other Web Engineering methodologies (cf. section 4.2).

4.1 Architectural Framework

In the following, we present a generic architecture serving as a framework for the technical realization of the ideas and concepts presented before. Figure 4 gives an overview of the architectural framework which was designed based on concepts from the fields of Service-oriented Architecture (SOA) and Enterprise Application Integration (EAI). Corresponding to the sphere concept presented in section 3, the architecture defines a *registry layer* for the semantic registry in the sphere's core, a *repository layer* for the ad-hoc and infrastructure repositories and a *client layer* for applications interacting with the Reuse Sphere.

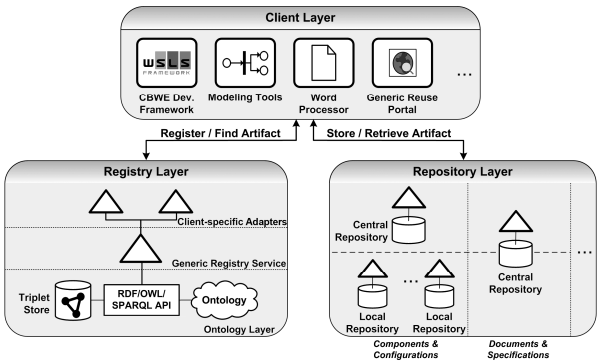

Figure 4. The architectural framework.

The *registry layer* comprises a Semantic Web API being able to deal with RDF, OWL and SPARQL, a triplet store for storing RDF instances and the Reuse Sphere's core ontology. Our current implementation uses the *Jena Semantic Web Framework* [1]. In order to allow for platform-independent storage and retrieval of RDF data as well as for executing SPARQL queries, we developed a CRUDS-based *Registry Web Service* on top of the Jena API. Thereby, clients can platform- and location-independently perform searches on the *triplet store* or create, read, update and delete metadata in form of RDF statements. Furthermore, the Registry Web Service is able to provide up-to-date information about the concepts and properties defined in the ontology, thus enabling applications to recognize new concepts or properties automatically and extend their metadata registration dialogs dynamically. Thereby, existing clients are invariant towards changes or extensions of the ontology.

As the Registry Web Service encapsulates the actual implementation of the Semantic Web API, any equivalent, possibly already existing framework could be integrated. On top of the Registry Web Service and following the Adapter design pattern [15], client-specific Web service *adapters* fulfilling specific interfaces required by particular client applications are located. Thereby, client applications providing mechanisms for external data source integration can communicate with the registry without requiring modifications to the client application itself.

Patrick Freudenstein, Marko Boettger, and Martin Nussbaumer: Efficacious Reuse Support as Enabler for Cross-Methodological Web Engineering with Stakeholders. In Proceedings of 8th International Conference on Web Engineering (ICWE2008), 14-18 July 2008, New York, USA.

The *repository layer* comprises all repositories on the ad-hoc (i.e. local repositories) and on the infrastructure level (i.e. central repositories), covering all types of artifacts, e.g. documents, components, models etc. In order to integrate these heterogeneous repositories into the Reuse Sphere, each of them is equipped with a dedicated Web service wrapper, thus leading to a homogeneous access layer for the distributed repositories. These wrappers share a uniform CRUDS-based interface, allowing for storing, retrieving, updating, deleting and searching (versioned) artifacts. Beyond that, repositories on the ad-hoc level are equipped with *observer* agents, being responsible for identifying new or modified artifacts, extracting metadata in accordance to the concepts and relations defined in the ontology and registering them with the registry. In our current implementation, we developed wrappers and observer agents for integrating file system-based repositories (mainly used on the ad-hoc level), Microsoft Office SharePoint Server 2007 repositories for all kinds of documents as well as the component and configuration store of our component-based Web Engineering framework, the WebComposition Service Linking System (WSLS) [13]. The file system wrapper, for example, could also be used for integrating file-based development frameworks and modeling tools from other Web Engineering methodologies.

The *client layer* covers all kinds of client applications participating in the Web Engineering Reuse Sphere by storing, registering, finding and retrieving artifacts. In order to integrate applications in the Reuse Sphere, we used the plug-in facilities provided by most of today's applications. As a first step, we developed plug-ins for the Microsoft Office suite including Microsoft Visio, the IBM Rational Software Architect and our WSLS framework (cf. section 5). These plug-ins provide dialogs for registering and storing artifacts to repositories on the infrastructure level as well as for finding and retrieving artifacts. Therefore, these dialogs communicate with the registry and repository Web services. Beyond that, we implemented a Web portal serving as central access point for interacting with the Reuse Sphere. This can be used if no client-specific plug-ins are available or for management operations by the Reuse Librarian.

4.2 The Ontology

The ontology forms the basis for registering and finding artifacts. So far, we included generic core concepts, properties and relations tailored to the Web Engineering reuse domain. A major design goal was to keep the ontology generic in order to allow for using it with any Web Engineering methodology, thus enabling cross-methodology reuse. We included well-defined extension points for integrating specific concepts and relations required for particular Web Engineering methodologies. In the following, we will present selected semantically cohesive parts of the ontology.

4.2.1 Knowledge and Stakeholders

Figure 5 shows a simplified excerpt of the concepts and relations covering the domains knowledge and stakeholders. These concepts are a central part of the ontology, as they are used to specify the semantic knowledge used for evaluating inference-based queries concerning the adequacy of artifacts, resolution strategies, modeling techniques and tools for given stakeholders. The white ellipses represent connecting concepts which are out of the current figure's scope.

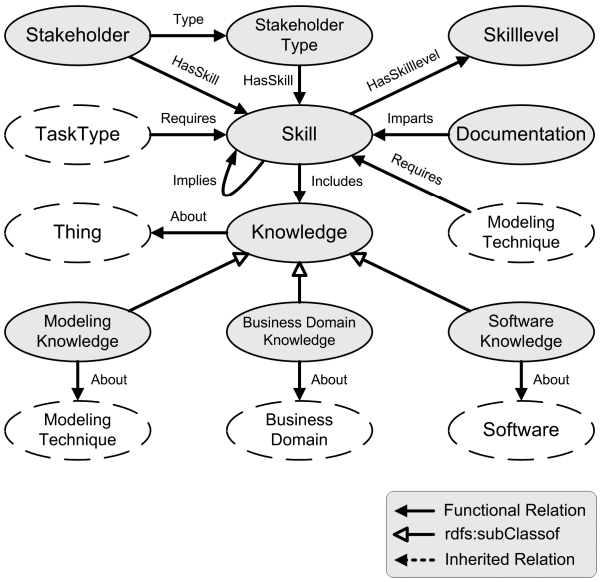

Figure 5: Knowledge & Stakeholders (Simplified Excerpt)

Therefore, the ontology includes the central concept *knowledge* which can be differentiated in several types of knowledge like *business domain knowledge*, *modeling knowledge* or *software knowledge*. The *about* relations between these knowledge types and the subjects of knowledge realize the connection to other concepts in the ontology, i.e. *modeling techniques*, *business domain* and *software*. The concept *skill* realizes the connection between *knowledge* and *stakeholders* or *stakeholder types* in the sense of having knowledge and with *task types* and *modeling techniques* in the sense of requiring knowledge. In each case, the relation is attributed with a *skill level* for

Patrick Freudenstein, Marko Boettger, and Martin Nussbaumer: Efficacious Reuse Support as Enabler for Cross-Methodological Web Engineering with Stakeholders. In Proceedings of 8th International Conference on Web Engineering (ICWE2008), 14-18 July 2008, New York, USA.

classifying the depth of the required or possessed *knowledge*. Furthermore, concepts and relations for expressing that a *documentation* can *impart* missing *skills* and that particular *skills imply* other *skills* are available.

4.2.2 Artifact, Methodology, Process and Product

A simplified excerpt of the concepts and relations around *artifact*, *Web Engineering methodology*, *process model* and *product* is depicted in Figure 6. This part of the ontology primarily provides the foundation for integrating Web Engineering methodologies along with their development process models, resolution strategies and artifact types.

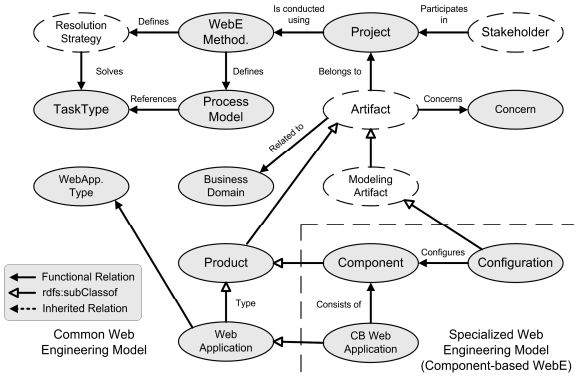

Figure 6: Artifact, Methodology, Process&Product (Simplified Excerpt)

By including instances of the concept *Web Engineering methodology*, methodologies like WebML, UWE, OOHDM or OO-H can be included in the ontology. Each methodology defines or refers to its software development *process model* which in turn refers to (ideally cross-methodologically shared) task types. Furthermore, each methodology defines one or more *resolution strategies* for every task type; this relation is considered in more detail in the next subsection. Naturally, the majority of tasks occurring in the development of a Web application, e.g. 'design workflows' or 'design navigation' can be found across all Web Engineering methodologies, even though their names differ amongst them. Thus, in order to support cross-methodological queries, referring to corresponding existing task types should always be preferred to defining new (redundant) task types.

Beyond that, concepts for diverse *artifact* types, e.g. *modeling artifact* or *product* are available. As indicated by the separate area in the figure, specific *artifact* types and *products* for each Web Engineering methodology can be integrated here. In the figure, extensions for a component-based Web Engineering methodology are shown, where *components* are subclasses of *product* which in turn is an *artifact*. Furthermore, they are configured with *configurations,* which in turn are a special type of *modeling artifacts*. Likewise, WebML could define subclasses or instances of the concept *modeling artifact* for its various model types, e.g. 'WebML Hypertext Model', 'WebML Business Process Model' etc.

The concept *project* is used to indicate in which project(s) an *artifact* was created or reused. Additionally, it can be expressed which *Web Engineering methodologies* were used in a *project*.

Artifact is the central concept in the ontology representing all kinds of reusable artifacts. It incorporates general metadata properties from the Dublin Core ontology and can be further classified with respect to related *project*(s), Web-specific *concern*(s) (e.g. Data, Navigation, Interaction, Presentation, Process etc.) or business domain (e.g. Travel Management, Procurement etc.).

4.2.3 Methodology-specific Resolution Strategies, Modeling Techniques and Tools

The integration of methodology-specific knowledge in the ontology is a crucial factor for cross-methodological reuse scenarios, e.g. determining *resolution strategies*, *modeling techniques* and *software* along with corresponding *artifacts* in accordance to a given *stakeholder*'s *skills* across various *Web Engineering methodologies*. Thereby, the strengths of each methodology can be used and, in combination with initiatives like the MDWEnet activity, the hitherto existing methodological frontiers be overcome.

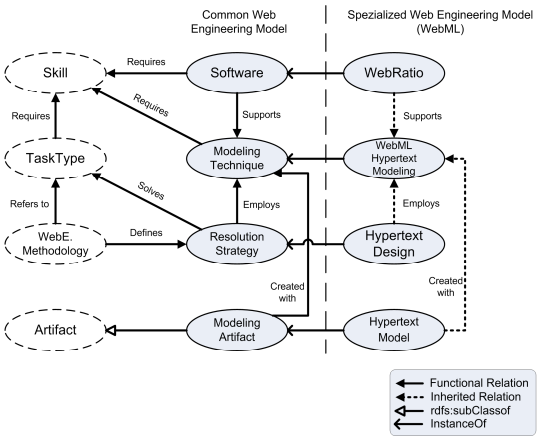

Figure 7: Extending the Ontology for WebML (Simplified Excerpt)

Figure 7 illustrates a simplified excerpt of the ontology covering the concepts *resolution strategy*, their *modeling technique*(s) and supporting *software* as

Patrick Freudenstein, Marko Boettger, and Martin Nussbaumer: Efficacious Reuse Support as Enabler for Cross-Methodological Web Engineering with Stakeholders. In Proceedings of 8th International Conference on Web Engineering (ICWE2008), 14-18 July 2008, New York, USA.

well as the resulting *modeling artifact*(s). As before, relations to connecting concepts are represented by white ellipses on the left. On the right side, dedicated instances for the *task type* 'Design Navigation' and the *Web Engineering methodology* 'WebML' were added following the 'Strategy' design pattern. Therefore, WebML proposes the *resolution strategy* 'Hypertext Design' that employs the 'WebML Hypertext Modeling' *modeling technique* which is supported by the *software* 'WebRatio' and results in the *modeling artifact* type 'Hypertext Model'.

Another example is shown in Figure 8 for a DSL-based Web Engineering approach [12, 22]. In this case, the extension is performed by adding new concepts for *DSL*, *Graphical Notation (DIM)*, *DIM Editor* etc. as subclasses of the core ontology's concepts. Based thereupon, instances for particular *DSLs* can be defined. This could be for example a Workflow *DSL* with *graphical notations* like BPMN, UML, Petri Nets and associated *editors*, e.g. Microsoft Visio, IBM Rational Software Architect or INCOME 2010. Such a Workflow DSL would be associated with the *task type* "Design Business Process" and the *Web Engineering methodology* 'DSL-based Web Engineering'.

Based on such methodology-specific ontology extensions, for a given task type and given stakeholder skills, suitable artifacts and resolution strategies can be cross-methodologically determined. Moreover, assumed that cross-methodological model interchange is possible as planned by the MDWEnet research activity, artifacts could be cross-methodologically found and even reused.

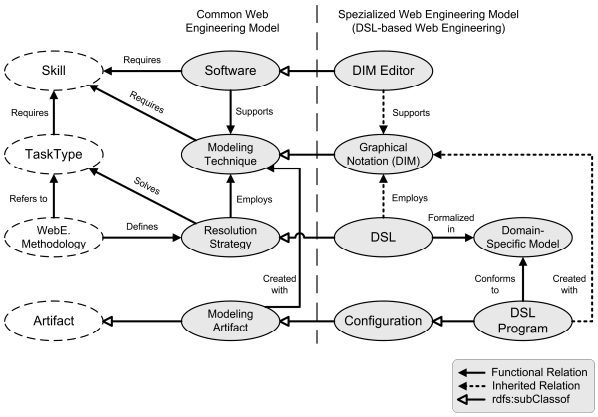

Figure 8: Extending the Ontology for DSL-based Web Engineering (Simplified Excerpt)

5. The Reuse Sphere Applied – an Example

In the following, we demonstrate how the Reuse Sphere approach considerably accelerates development speed and improves stakeholder communications in the Web Engineering domain. The example scenario stems from a university-wide EAI project [17] and deals with the development of a workflow-based feature supporting the master thesis business process for all involved parties within a Web portal. As consolidation and interoperability activities like *MDWEnet* are still in progress, we can only point out how cross-methodological reuse could be achieved in the particular steps. Nonetheless, we show how the Reuse Sphere approach can significantly improve and ease reuse in today's Web Engineering methodologies.

As mentioned in section 4, we integrated – amongst others - the WebComposition Service Linking System (WSLS) in the Reuse Sphere's architectural framework. WSLS is a development framework for component-based Web Engineering, i.e. it allows for constructing Web applications by assembling and configuring components. As configuring a component is a non-trivial task, especially for stakeholders without software development skills, we introduced the idea of using XML-based DSL programs for the configuration. In earlier papers, e.g. [12, 22], we presented DSLs for various aspects within a Web application, e.g. workflows, dialogs, application structure etc. A DSL provides diverse graphical notations, each of them being tailored to a particular stakeholder group and focusing simplicity. Thus, Web applications can be constructed by assembling components and configuring them with DSL programs which are modeled using stakeholder-specific graphical notations and tools.

However, as shown in section 4, our DSL-based approach is – as well as other Web Engineering methodologies – only a specific extension of the presented ontology. Likewise, it was shown how other Web Engineering methodologies can be incorporated. Thus, from the perspective of the Reuse Sphere approach, the following examples can be directly transferred to other Web Engineering methodologies, thereby not restricting the achieved improvements to the DSL-approach.

In the first scenario, we consider the realization of the master thesis business process as Web-based workflow feature. Thereby, a great variety of stakeholder types with diverse skills is involved in its conceptual design, e.g. staff from the library, the exam office, students, advisors and professors. In the following example, we assume that a repository search for existing master thesis workflows had no results. Thus, we want to start the conceptual design with a representative of the advisors stakeholder group. Therefore, a suitable resolution strategy for this task has to be determined first. We open the registry search

Patrick Freudenstein, Marko Boettger, and Martin Nussbaumer: Efficacious Reuse Support as Enabler for Cross-Methodological Web Engineering with Stakeholders. In Proceedings of 8th International Conference on Web Engineering (ICWE2008), 14-18 July 2008, New York, USA.

dialog within WSLS and choose the search strategy 'Search for Resolution Strategy and related Artifacts' as depicted in Figure 9-1. In the next dialog, we select the current process phase 'Conceptual design' and thereupon the task type we want to perform, i.e. 'Design Business Process'. Based on that, the registry Web service is called which configures a predefined SPARQL query template with the given task type and executes it, resulting in a set of possible resolution strategies, modeling techniques, software and their required skills. Based on that, the third dialog is constructed. There, we can either select a predefined skill set corresponding to the given stakeholder type or specify a skill level for each knowledge type.

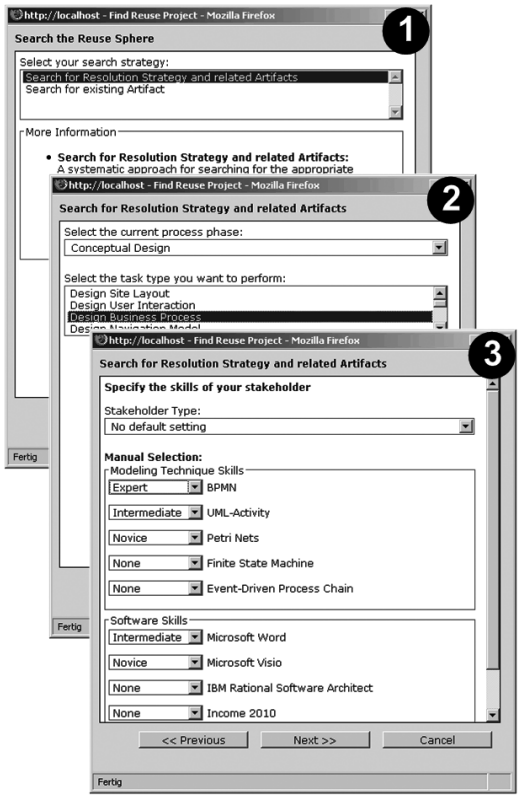

Figure 9: 'Search for resolution strategy' wizard

The selected skills are submitted to the Registry Web service which thereupon configures a predefined SPARQL query template and executes it. The query results in a list of matching resolution strategies, modeling techniques and software as well as related artifacts. Thereby, statements from the RDF triplet store expressing that particular skills imply other skills or that a documentation can impart missing skills are also evaluated. Finally, the results are ranked according to the matching degree between the specified and inferred skills and the required skills.

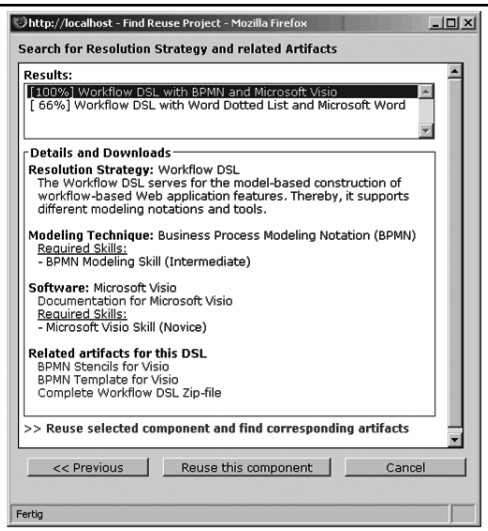

Figure 10: Search results

Figure 10 shows the search result dialog. The resolution strategy 'Workflow DSL' with the modeling technique 'BPMN' and supported by the software 'Microsoft Visio' was identified as a perfect match for the given stakeholder. In details pane, the individual elements are listed along with their required skills. As the stakeholder stated only 'Novice' knowledge in Microsoft Visio, a link to a documentation is provided. Moreover, download links for related artifacts (e.g. a Microsoft Visio template for starting the modeling of the workflow) for the selected result are listed. By clicking on 'Reuse this component', the software component associated with the selected result is inserted in the current WSLS development project. It has to be configured with a DSL program which could either be modeled using the downloadable template or searched for by following the link 'Reuse selected component and find corresponding artifacts'.

In the context of *cross-methodological reuse*, the result set would comprise additional resolution strategies, modeling techniques and tools from other Web Engineering methodologies as well as related artifacts. For example, if WebML was integrated in the ontology as indicated in Figure 3, the result 'WebML Business Process Design' with the modeling technique 'WebML Process Modeling with BPMN' and supported by the software 'WebRatio' along with appropriate templates or documentation would be included.

The second scenario deals with the realization of a dialog within the master thesis workflow. The dialog should be used by students to apply for a vacant master thesis. As many chairs at our university have already implemented such dialogs on their homepages, we want to reuse an existing dialog and adapt it to the

Patrick Freudenstein, Marko Boettger, and Martin Nussbaumer: Efficacious Reuse Support as Enabler for Cross-Methodological Web Engineering with Stakeholders. In Proceedings of 8th International Conference on Web Engineering (ICWE2008), 14-18 July 2008, New York, USA.

given requirements. Thereby, we collaborate with a representative from the students stakeholder group.

Within WSLS, we open the registry search dialog and select the search strategy 'Search for existing Artifact' (Figure 11). In the next dialog, parameters for the search query can be selected. According to the given scenario, we search for a 'DSL Program' related to the business domain 'Advising' and the concern 'Interaction' that can be used for the task type 'Design Dialog'. By selecting the given stakeholder type or the related modeling technique, we could already constrain the query according to the knowledge required for the modification of a found artifact. Beyond that, keywords could be provided for a full-text search.

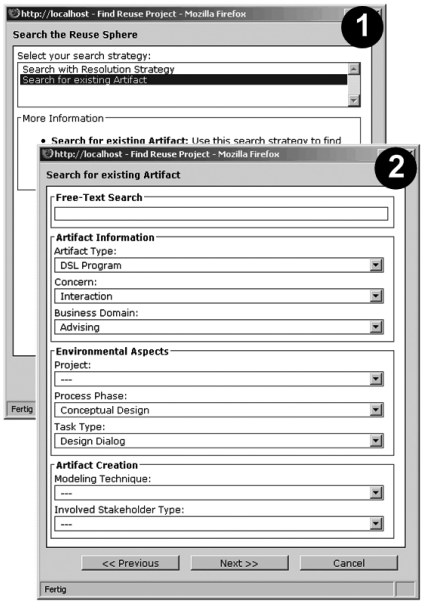

Figure 11: 'Search for existing Artifact' wizard

The query parameters are again submitted to the Registry Web service which inserts them in a predefined SPARQL template and executes it. The returned results are shown in Figure 12. This result set covers both the sphere's ad-hoc and infrastructure levels and can now be further refined by browsing through the registry space. Therefore, all values in the details pane, e.g. the projects the artifact was used in, are rendered as hyperlinks allowing for a context switch. Moreover, the result set can be filtered by selecting the given stakeholder's modeling and software knowledge. Both browsing and filtering are realized by executing SPARQL queries via the Registry Web service. The 'Preview' button allows for testing the selected DSL program together with its executing component in the current WSLS development project, thus easing communication with the stakeholder.

As described for the previous scenario, artifacts from other methodologies could also be found here, e.g. WebML or UWE dialog models. This could be achieved without any modifications to the SPARQL query or the dialogs. Assumed that dialog models were interoperable as strived for by the *MDWEnet* initiative, they could even be directly reused.

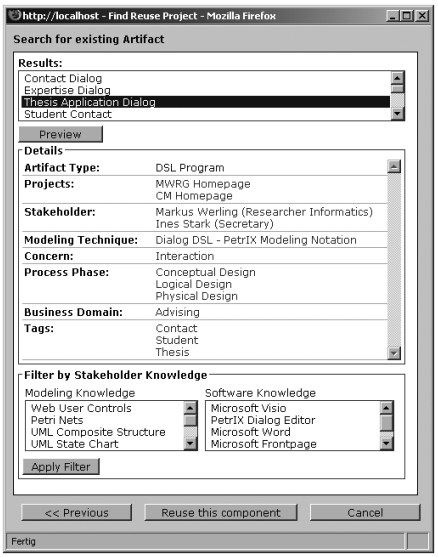

Figure 12: Results with browsing, filtering and preview facilities

6. Related Work

In the Web Engineering research field, reuse-related research primarily focuses the adequacy of models and software components for reuse. For example, in [27], Schwabe et al. introduce the concept of 'OOHDM Frames', i.e. Web design frameworks for specifying common design schemas and their variation points, thus fostering reuse on design level. A similar idea called 'WebML skeletons' is presented in [6]. Such skeletons specify abstract and simplified versions of recurring structural and hypertext schemas for being instantiated and reused.

In conclusion, these approaches provide interesting insights on how reuse on a model level could be improved by identifying and modeling recurring abstractions and reusing them by instantiation for a particular application. Already here, the similar goal of both approaches awakes the desire for a unifying, cross-methodological reuse approach for models.

With respect to reusing Web components and their code, the WebComposition approach presents its dedicated WebComposition Repository in [14]. It aims at facilitating the storage and retrieval of components, thereby allowing for incorporating various metadata

Patrick Freudenstein, Marko Boettger, and Martin Nussbaumer: Efficacious Reuse Support as Enabler for Cross-Methodological Web Engineering with Stakeholders. In Proceedings of 8th International Conference on Web Engineering (ICWE2008), 14-18 July 2008, New York, USA.

representation methods as postulated by Frakes and Pole in [9]. Efficiently finding reusable components and code is a key factor, not only for Component-based Web Engineering, but also for other Web Engineering methodologies. Thus, it seems desirable to have a methodology-independent reuse approach establishing a basis for reuse both on model and component level.

In [7], the authors present the 'Kuaba Ontology' - an inspiring ontology-based approach for reusing Design Rationales, i.e. the reasons and justifications for design decision, and associated artifacts. Although this problem domain is different from ours, the idea of establishing a unifying, methodology-independent foundation in form of an ontology is similar to our approach.

Beyond that, the Web Engineering community currently strives for realizing the hitherto untapped potential of interoperability and model interchange across today's Web Engineering methodologies. In [28], a generic framework defining a common denominator and enabling the comparability of these methods is proposed. Such research is a vital step for achieving interoperability and thus also an important input for the presented Reuse Sphere approach. Beyond that, consolidation efforts like the Model-driven Web Engineering initiative *MDWEnet* [32] strive for achieving practical interoperability between common model-driven Web Engineering methodologies. Thus, the potential of the presented Reuse Sphere approach becomes even more obvious, as it is not only applicable across today's Web Engineering methods, but also enables real cross-methodological reuse.

7. Conclusion & Future Work

Facing the need for a comprehensive reuse concept tailored to the Web Engineering domain, we presented the Web Engineering Reuse Sphere approach. With regard to current consolidation efforts towards interoperability of today's Web Engineering methodologies, e.g. *MDWEnet*, the Reuse Sphere establishes a common foundation for real cross-methodological reuse.

We presented the Reuse Sphere's concept of a distributed, cross-methodological repository space consisting of two spheres for spontaneous and planned reuse. A central ontology-based registry in the Reuse Sphere's core registers all semantic metadata and provides holistic registration and search functionalities. The presented ontology provides well-defined extension points for other Web Engineering methodologies and allows for efficacious, cross-methodological searches. Thereby, stakeholder characteristics represent an integral context parameter for inference-based searches. Thus, artifacts can be found according to the skills required for their validation, modification or usage as templates for new artifacts.

Beyond that, we presented an architectural framework for the technical realization of the Reuse Sphere. Thereby, we explained how existing (local and infrastructure-based) repositories and clients from other Web Engineering methodologies can efficiently and efficaciously be integrated.

Throughout the paper, we pointed out the Reuse Sphere's extensibility and applicability for cross-methodological Web Engineering and demonstrated how other methodologies can be incorporated. Based on an example scenario from practice, we demonstrated how the presented approach considerably accelerates development speed and improves stakeholder communications in the Web Engineering domain.

Our future work focuses on the continuous alignment with consolidation activities like *MDWEnet*, thus assuring mutual benefits. Moreover, we are planning to investigate the automatic derivation of metadata from an artifact's context or a user's behavior while working with it in more detail. Beyond that, an alignment with the Kuaba Ontology approach [7] mentioned in the previous section seems to be promising. Thus, based on Design Rationales, even more support for selecting an appropriate resolution strategy, modeling technique or software could be provided. Furthermore, stakeholders could be assisted by evaluating experiences gained by other stakeholders.

8. Availability

The presented ontology as well as the WSLS framework can be downloaded from our research homepage http://research.tm.uka.de.

9. References

1. Homepage of the Jena Semantic Web Framework - 2003): http://jena.sourceforge.net/ (13.02.2008)
2. Dublin Core Metadata Initiative RDF Schemas - 2008): http://dublincore.org/schemas/rdfs/ (12.02.2008)
3. Boldyreff, C., Nutter, D., and Rank, S.: Active Artefact Management for Distributed Software Engineering. in Proc. of the 26th International Computer Software and Applications Conference on Prolonging Software Life:

Patrick Freudenstein, Marko Boettger, and Martin Nussbaumer: Efficacious Reuse Support as Enabler for Cross-Methodological Web Engineering with Stakeholders. In Proceedings of 8th International Conference on Web Engineering (ICWE2008), 14-18 July 2008, New York, USA.

Development and Redevelopment. 2002: IEEE Computer Society
4. Brickley, D. and Miller, L.: FOAF Vocabulary Specification 0.91 - 2007): http://xmlns.com/foaf/spec/ (12.02.2008)
5. Ceri, S., Fraternali, P., and Bongio, A.: Web Modeling Language (WebML): A Modeling Language for Designing Web Sites. in 9th International World Wide Web Conference (WWW). 2000. Amsterdam, Nethderlands
6. Ceri, S., Fraternali, P., and Matera, M.: WebML Application Frameworks: a Conceptual Tool for Enhancing Design Reuse. in WWW10 Workshop Web Engineering. 2001. Hong Kong
7. De Medeiros, A.P., Schwabe, D., and Feijo, B.: Kuaba Ontology : Design rationale representation and reuse in model-based designs. in Proc. of 24th International Conference on Conceptual Modeling. 2005. Klagenfurt, Austria
8. Fensel, D.: The OntoWeb Ontology Homepage - 2003): http://www.ontoweb.org/Ontology/index.html (08.02.2008)
9. Frakes, W.B. and Pole, T.P.: An Empirical Study of Representation Methods for Reusable Sofware Components. IEEE Transactions on Software Engineering, 1994. 20(8): p. 617
10. Fransen, J.: Customizing the Microsoft Office 2003 Research Task Pane - 2003): http://msdn2.microsoft.com/en-us/library/aa159647(office.11).aspx (12.02.2008)
11. Freeman, P.: Reusable Software Engineering: Concepts and research directions. in ITT Proceedings of the Workshop on Reusability in Programming. 1983. Newport, RI, USA
12. Freudenstein, P., et al.: Model-driven Construction of Workflow-based Web Applications with Domain-specific Languages. in Proceedings of the 3rd International Workshop on Model-Driven Web Engineering (MDWE 2007). 2007. Como, Italy: CEUR Workshop Proceedings, ISSN 1613-0073.
13. Gaedke, M., Nussbaumer, M., and Meinecke, J.: WSLS: An Agile System Facilitating the Production of Service-Oriented Web Applications, in Engineering Advanced Web Applications, S.C. M. Matera, Editor. 2005, Rinton Press. p. 26-37
14. Gaedke, M. and Rehse, J.: Supporting Compositional Reuse in Component-Based Web Engineering. in 2000 ACM Symposium on Applied Computing (SAC 2000). 2000. Villa Olmo, Como, Italy: ACM
15. Gamma, E., et al.: Design patterns: elements of reusable object-oriented software. Addison-Wesley professional computing series. 1995, Reading, Mass.: Addison-Wesley. xv, 395
16. Gómez, J. and Cachero, C.: OO-H Method: Extending UML to Model Web Interfaces, in Information modeling for internet applications, P.v. Bommel, Editor. 2003, IGI Publishing: Hershey, PA, USA. p. 144 - 173
17. Juling, W.: KIM Project Homepage - 2005), University of Karlsruhe: http://www.kim.uni-karlsruhe.de/ (24.04.2005)
18. Koch, N., et al.: UML-Based Web Engineering : An Approach Based on Standards, in Web Engineering: Modelling and Implementing Web Applications, G. Rossi, et al., Editors. 2007, Springer: London. p. 157-191
19. Krueger, C.W.: Software reuse. ACM Computing Surveys, 1992. 24(131): p. 131-183
20. Mcconnell, S.: Chapter 33: Reuse, in Rapid Development. 1996, Microsoft Press: Redmond, Washington, USA. p. 527-538
21. Mcllroy, M.D.: Mass Produced Software Components. in Sofiware Engineering; Report on a conference by the NATO Science Committee. 1968. Garmisch, Germany: NATO Scientific Affairs Division, Brussels, Belgium
22. Nussbaumer, M., Freudenstein, P., and Gaedke, M.: The Impact of DSLs for Assembling Web Applications. Engineering Letters, 2006. 13(2006): p. 387-396
23. Nussbaumer, M., Freudenstein, P., and Gaedke, M.: Stakeholder Collaboration - From Conversation To Contribution. in 6. International Conference on Web Engineering (ICWE). 2006. SLAC, Menlo Park, California: ACM
24. Prieto-Diaz, R.: A faceted approach to building ontologies. in IEEE International Conference on Information Reuse and Integration. 2003. Las Vegas, USA
25. Prud'hommeaux, E. and Seaborne, A.: SPARQL Query Language for RDF - W3C Recommendation (2008), World Wide Web Consortium (W3C): http://www.w3.org/TR/rdf-sparql-query/
26. Roger S. Pressman: Part Three: Applying Web Engineering, in Software Engineering: A Practitioner's Approach. 2005, McGraw-Hill: New York. p. 499-626
27. Schwabe, D., et al.: Engineering Web applications for Reuse. IEEE Multimedia, 2001. 8(1): p. 20-31
28. Semia Sonia Selmi, Naoufel Kraiem, and Ghezala, H.B.: Toward a Comprehension View of Web Engineering. in 5th International Conference of Web Engineering (ICWE 2005). 2005. Sydney, Australia: Springer
29. The Standish Group International: CHAOS Research - Research Reports (1994-2005): http://www.standishgroup.com
30. Tracz, W.: Where does reuse start? ACM SIGSOFT Software Engineering Notes, 1990. 15(2): p. 42-46
31. Uschold, M. and King, M.: Towards a Methodology for Building Ontologies. in Workshop on Basic Ontological Issues in Knowledge Sharing. 1995. Montreal, Canada
32. Vallecillo, A., et al.: MDWEnet: A Practical Approach to Achieving Interoperability of Model-Driven Web Engineering Methods. in Third International Workshop on Model-Driven Web Engineering (MDWE'07). 2007. Como, Italy
33. Yongbeom, K. and Edward, A.S.: Software Reuse: Survey and Research Directions. Journal of Management Information Systems, 1998. 14(4): p. 113-147