Taxonomies in Operation, Design, and Meta-Design

Claudia Niederée, Claudio Muscogiuri, Matthias Hemmje
FhG - IPSI
D-64293 Darmstadt, Germany
[niederee;muscogiuri;hemmje]@ipsi.fhg.de

Abstract

Taxonomies are a well-established instrument for organizing and accessing resources in Information, Content and Knowledge Management (ICKM) systems. Furthermore, they contribute to a common understanding and an improved communication in the user community by fostering the development and usage of a shared vocabulary.

In addition to these operational usage scenarios, we argue in this paper that a taxonomy is also a valuable medium in system design. We present a meta-design framework for systematically supporting the user in the setup, customization and evolution of We-based ICKM system instances, which is based on a model-based domain construction approach. Taxonomies are exploited both as the fundamental basis for the construction process itself and as the principal support for context-driven access to the common metamodel in this framework implementing the ontological commitment underlying the complete framework.

These manifold forms of taxonomy exploitation are supported by a flexible taxonomy component that enables multiple classifications of arbitrary resources, effective taxonomy management, and context-adaptive taxonomy reduction in our ICKM meta-design framework.

1. Introduction

Classification is a well-established mechanism in the area of organizing Information, Content, and Knowledge resources (*ICK* resources). When classification is not restricted to unrelated keywords, taxonomies come into play establishing hierarchical “is a” relationships between the concepts used for classification. The development and spreading of IT technologies like hypertext-based approaches, machine learning, etc. opened new application areas for taxonomies, because they added mechanisms for easy navigation as well as for automatic handling and intelligent exploitation to the existing power and attractiveness of taxonomies. Examples of modern taxonomy-centered applications are Web catalogues like Yahoo![1]. Currently, taxonomies have another renaissance due to their role as core technology for the realization of ontologies in the Semantic Web context [1].

In this paper we consider taxonomy exploitation in the context of a framework for Web-based Information, Content, and Knowledge Management (*ICKM*) systems. Example ICKM systems are e-learning and qualification management systems, which are dedicated to increasing knowledge and managing ICK resources like courses, tests, student profiles and qualification goals and e-commerce systems like trade fair support systems that aim at supporting the trade fair business process and manage ICK resources like stand and product descriptions, user profiles, and marketing information [21]. The usefulness of a taxonomy in an ICKM system is, however, not restricted to the management of the ICK resources, but can also be applied to the system design process.

For supporting a wide variety of ICKM systems a flexible framework is required, which is adaptable to the organizational content and knowledge processes, which themselves are not static, but evolve dynamically over time. We believe, that in such a framework a wide range of evolutionary changes can be supported in the system itself by empowering the users to participate in system setup, customization, and evolution bringing their domain expertise into the system. This approach follows the idea of meta-design as it is described in [2]: the framework provides integrated support, in our case system authoring tools, for participating in its own design process.

In our ICKM meta-design framework we, thus, have two types of components:

- *ICKM components:* An extensible set of ICKM components provides content and knowledge management functionalities for the effective acquisition, enrichment, retrieval, and context-adaptive dissemination of ICK resources.
- *System Authoring tools:* The Web application development process incorporates entities and concepts from different domains and on different levels of granularity like hypertext navigation maps,

[1] http://www.yahoo.com/

business process steps, Web pages, form fields, entities from the application domain, user tasks, etc., which have to be set into relationship to each other during the design and development process. User tasks, for example, have to be mapped to business steps and Web pages have to be put in the context of a navigation map. For this purpose the framework contains an extensible suite of general as well as task-specific system authoring tools, which are used to step-by-step set up and customize the complete Web application model for the ICKM system instance under construction based on the ICKM components and on heuristics and templates for effective ICKM system setup.

Classifying the design-related system objects by a common taxonomy provides a common conceptual basis for the tools facilitating the mapping between entities and the cooperation between tools and contributes to a common understanding between the different stakeholders involved into the ICKM system development process. The use of the same paradigm, namely taxonomy and classification, for operation, design and meta-design facilitates user empowerment with respect to system design and evolution by stressing the commonalities and synergies between the different processes.

In the ICKM meta-design framework, we present in this paper, taxonomies are exploited in various ways differing in the usage domain (content management, system design, etc.), the visibility of the usage to the system user (explicit and implicit usage), and in the intension of taxonomy usage (information navigation, design support, etc.). Capturing the various different taxonomy usage dimensions results in a complex, manifold taxonomy underlying the ICKM meta-design framework. Avoiding information overload and fostering work focus, users are not confronted with the entire taxonomy, but only with the part that is relevant for their current *context of usage*. For this purpose taxonomies are reduced dynamically driven by a model of the current working context (*context-adaptive taxonomy views*). Since we operate in a meta-design environment the framework does not only have to provide components and user interfaces for taxonomy navigation and management. We also need tools to set up dialog sequences that support business steps and involve the different forms of taxonomy usage, which adds an extra layer of abstraction. Especially, we need (task-specific) tools that enable the application author to control taxonomy reduction while setting up application pages.

The rest of the paper is structured as follows: After summarizing the ideas of meta-design and user empowerment, section 2 discusses the challenges and opportunities in Web applications design, development, and evolution. Section 3 explains the conceptual model for implementing an ICKM meta-design framework and of exploiting the benefits of taxonomies in meta-design. Section 4 presents the classification model underlying our approach and the idea of context-adaptive taxonomy views together with the context-driven taxonomy reduction process. Section 5 describes the developed ICKM meta-design framework focusing on the taxonomy management component and its interaction with the rest of the framework components. The paper concludes with an outlook to future research plans in the context of the framework.

2. State of the Art and Related Work in Supporting Web Applications Development

2.1. Web Application Development and Evolution

The development of a large software system is a complex process that in general requires several iterations until the produced system fulfils the requirements of the intended user community [3]. This is especially true for Web applications implementing e-solutions, for which the requirement analysis and design phases necessitate the involvement of a variety of stakeholders [4], including, but not necessarily limited to marketing people, consultants, creative designers, customer support, vendors, lawyers, business executives, and other domain experts as well as system designers and developers.

Even when the developed system has reached a state upon which all stakeholders agree and is finally deployed, the system change process does not stop. Business process redesign due to new user requirements, user feedback, new legal regulations, competitive reasons, etc. may require an adaptation of the associated Web application to the new process. Because of the particularly dynamic global scenario in which the e-solution is positioned, the requirements toward the system change as frequently as the worldwide markets and communities' needs and trends do.

Designing and implementing a new system also contributes to the creation of a shared understanding between all the stakeholders. Once the users find themselves working together using the same system and exploiting this shared understanding, this can lead to "new insights, new ideas, and new artifacts" [2]. This provides an important additional source for system evolution requests.

System evolution, therefore, is an integral part of the operational phase of a Web application's lifecycle and has to be supported efficiently. The traditional software development cycle delays the integration of change requirements coming up in the operational phase to the next software release. This delay can be unacceptable in the highly competitive market of e-business.

As an alternative or complement to this software evolution approach the user can be empowered to implement at least certain classes of changes by himself.

2.2. User Empowerment

In fact, we believe, that a wide range of evolutionary changes can be supported in the Web-based ICKM system framework itself by providing adaptation support mechanisms. For this purpose, meta-design [2] for Web applications is required, i.e., the Web-based system has to provide integrated support for participating in its own design process. To achieve this goal, our framework provides powerful Web application authoring support: it empowers special users of the Web based system, the power users [5], not only to use the system but also to adapt it to changing requirements.

For real user empowerment we are interested in system modification that goes beyond user interface customization. Users should be enabled to manipulate system functionalities to adapt it to the specific business process. Such modifications are normally reserved to skilled software developers. Since the power users are members of the application domain, we may not assume programming skills.

It has to be noted here, that we do not expect power users to develop an entire ICKM solution from scratch on their own. The focus is on small incremental changes and adaptations in an operational system, which already supplies modular, task-specific packages for the support of the different tasks and roles in the business process, like the ICKM components.

If we want to convert at least part of the users from normal users into power users that act as co-designers and developers in system customization and evolution, adequate task-specific design support is required for these users. It is the aim of our approach to provide a framework for this kind of design support.

2.2.1 Model-based Development Paradigm

An extended analysis of the designers' activities and their tools [6] confirms that design communities evolve through a domain construction process. Within the context of a ICKM meta-design framework such a domain construction process is performed on a particular domain that is indeed a complex syntactic and semantic coalescence of objects and relationships between them, both coming from two originally separated domains:

- the domain of Web application development,
- the application domain, which includes business objects, relationships and business processes of the specific ICKM application domain.

This requires the ability of models to systematically specify the process of mapping between objects of the Web application user interface to business objects or business process subtasks of the application domain.

The validity of such approach is confirmed by analyzing best practice in the UI designer community [7]. For instance, following the so-called model-based interface development paradigm [8], model-based UI tools are used to support the design and the development of UIs in a professional and systematic way allowing designers/developers to specify UI design objects, like hypertext navigation maps, Web pages, form fields, by a) managing entities and relationship in the following domains: domain model, user model, presentation model, dialog model, task model, and b) setting mapping properties among entities from the different domains. The study of such design environments confirm that the model-based paradigm is a good fit for Web-based UI that rely on the use of modular components

So, following such a model-based paradigm, what we want is to allow power users of the ICKM solution to be engaged in design activities in the context of Web application authoring (development), supported by a suitable model-based framework.

2.3. Task-specific Design Support

There is an analogy between the design activities in the context of application authoring and end-user programming activities; thus, the challenge for the ICKM meta-design framework is to provide a highly specialized end-user programming environment that leverage users' existing *task-related* interest and skills. Nardi's studies on end-user programming [5] confirm that formal languages for end users must be task-specific programming languages:

"…it's only when people have a particular interest in something, that they readily learn the formal languages and notations that describe the <u>elements</u> and <u>relations</u> of the system of interest… People are likely to be better at learning and using computer languages that closely match their interest and their domain knowledge"

Therefore, ICKM meta-design framework must provide *task-specific* design support. Having users engaged in design activities, in analogy to end user programming activities, means involving users in a formal communication with the design environment.

Eventually, users/designers involved in such communication will want to communicate constraints, opportunities, knowledge about goals needed to be achieved, intentions (specific actions to take the goal), executions (specifics of the actions to performed) [9]; knowledge that leads the design of needed application logic; knowledge that will lead the design of the Web application. Such knowledge defines the context for the formal communication in the design process; therefore, we call the process of defining and designing such knowledge "context definition". For instance, workflow models can provide context definition to be used in designing task models, considering that low-level workflow activities may correspond to high-level tasks to be performed [23].

The implemented approach to task-specific design support is also influenced by the idea of DODEs (Domain-oriented Design Environments) [24]. Essentially, a domain oriented design environment includes a construction kit providing a palette of domain building blocks, an argumentative support, a catalogue consisting of a collection of pre-stored designs, a specification component supporting the interaction among stakeholder, and a simulation component (carrying out "what-if" games).

It is our aim to build a design environment that is not restricted to one application domain but usable for an entire class of systems, namely ICKM solutions with community support in different application domains like trade fair business, e-learning, etc. For this purpose, we generalize the DODE approach by factoring domain-specific knowledge into a domain model and developing a meta-model for the interaction of the domain model with the components of the design environment.

3. The conceptual model for the ICKM meta-design framework

In order for the ICKM meta-design framework to take advantage of task-specific user knowledge the model-based development paradigm has to be extended by integrating as design products different models that are suitable for the context definition. Figure 1 shows the conceptual model for the ICKM meta-design framework. In the depicted conceptual model the model-based paradigm is extended by integrating the models needed for generic Web application design (presentation, dialog, task, user and domain models) with a business workflow model and task-specific context models. In fact, as already mentioned, the business workflow model can provide context definition that can be used for the design of task models, as well as for user models and domain models: For instance low-level business workflow activities can be considered as action template parameters for a task-hierarchy in terms of needed resources upon which to operate, pre- and post-conditions, and other contextual information. Additionally, domain experts can provide task-specific context models for specific task-model.

Figure 1 also shows some examples of heuristics models (usability model and quality management model). The exploitation of heuristics models will be described in more detail in future publications; here we can say that processes in apparently all domains are driven by heuristics and best practices, which are based on individual and community experiences with the same or similar processes.

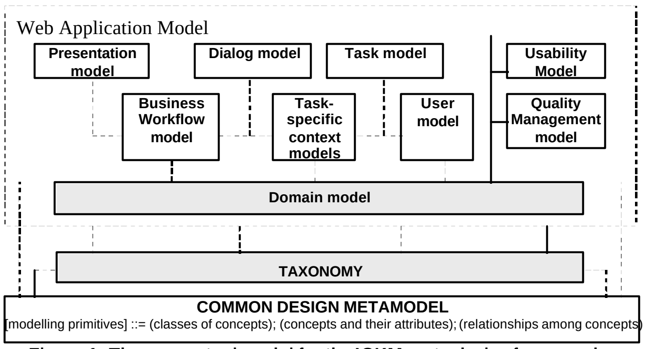

Figure 1: The conceptual model for the ICKM meta-design framework

The conceptual model, which is depicted in Fig. 1, is implemented in the ICKM meta-design framework, where the underlying common design metamodel supports the design activities in the ICKM meta-design framework: Publishing tools (for generic Web application design), context definition tools, heuristic components (and the users themselves) share the same metamodel in terms of modeling primitives (classes of concepts, concepts and their attributes, relationships among concepts), while contributing to the Web application metamodel that embraces the different involved models in the ICKM system (see Fig. 4 in chapter 4).

3.1. Taxonomy Exploitation in Meta-Design

As explained in the previous section, in the ICKM meta-design framework several tools and component as well as users, share the same design metamodel and contribute to create models that describes different classes of objects, properties of objects, and relations between objects across different domains. All of them, then, contribute to the definition and the construction of a shared composite model, the Web application model, by using a shared vocabulary, represented by the common design metamodel (see Fig. 1).

More formally, design activities in the ICKM meta-design framework aim athe dynamic creation, construction and evolution of an ontology, accordingly to Gruber's definition: "an ontology is a formal, explicit specification of a shared conceptualization" [10]. In the context of the ICKM meta-design framework, the *conceptualization* is the Web application model. On the operation level, the Web application model, as a product of a common effort of conceptualization, can be considered an ontology, where an ontology is "an engineering artifact, constituted by a specific vocabulary used to describe a certain reality, plus a set of explicit assumptions regarding the intended meaning of the vocabulary words. … In the simplest case, an ontology describes a hierarchy of concepts related by subsumption relationships“ [11], i.e., a taxonomy.

Ontologies generally appear as a taxonomic tree of conceptualizations, from very general and domain-independent at the top levels to increasingly domain-specific further down in the hierarchy [12].

In defining the Web application model, many taxonomy subtrees will be created and instantiated, all in the common taxonomy that underlies the ICKM meta-design framework (see Fig. 1), depending on the class of entities the modeler from the different domains commit to design for.

For the model-based domain construction process in the ICKM meta-design framework, taxonomies are, thus, exploited both as the fundamental basis for the construction process itself and as the principal support for context-driven access to the common metamodel:

1. Taxonomies facilitate the creation of a common classification schema for organizing and classifying entities/information objects from the different domain models (like user profiles, workflow steps, menus, etc.), thus implementing the ontological commitment underlying the complete framework of generic Web design and task-specific tools; where the ontological commitments is the agreements to use a shared vocabulary in a coherent and consistent manner with respect to the content theory specified by the shared conceptualization [13].
2. Taxonomies offer a consistent medium for accessing the common metamodel within the framework as well as querying and browsing the ICK resources. In fact, generic Web design as well as task-specific authoring tools access entities according to specialized views on the common metamodel. Taxonomies enable the definition of the specific “context of usage” that is propaedeutic to defining the specialized views.

Taxonomy usage exploitation in the ICKM meta-design framework is governed by generic usage patterns that come up in different variants depending on the type of the classified resources and the ICK process context. A flexible taxonomy component that enables multiple classifications of arbitrary resources, effective taxonomy management, and context-adaptive taxonomy reduction is a prerequisite for this type of taxonomy exploitation.

4. Context-adaptive Taxonomy Views

Capturing the various different taxonomy usage dimensions in content management and Web application design results in a complex, manifold taxonomy underlying the ICKM meta-design framework. Users are, however, not confronted with the entire taxonomy, but only with the part that is relevant for their current *context of usage*. We use the term *context-adaptive taxonomy views* for this kind of reduced taxonomies.

The taxonomy is used to classify information objects. In our approach, taxonomy management and reduction is based on a classification model that supports multiple classification and typed classification relationships. Taxonomy reduction is based on context modeling and makes use of functional taxonomy subtrees, which reflect common pattern of taxonomy structuring in a ICKM meta-design framework. It has to be pointed out that due to our meta-design approach we do not only need mechanisms for reducing taxonomies according to the context of usage, but also (task-specific) tools that enable the power user to control taxonomy reduction while setting up Web application pages, which are part of the Web application model.

4.1. Scenarios of Taxonomy Usage

Before we discuss taxonomy usage scenarios we will have a look on the different types of users involved in these scenarios. Typically, we can distinguish two groups of users for ICKM systems. On the one hand, there is a group of users managing and operating the system, like, e.g., trade fair organizers, qualification portal providers, etc. In a meta-design framework these users are involved in system setup and evolution as well as in content management. On the other hand, there are the users of the ICKM system instance, which are consuming the offered content and services. They are the final end users of the ICKM system instance. To distinguish these two groups we will refer to the second group as end users whereas the first group will be called power users, although they can also be considered as end users with respect to the system authoring and administration tools. We use the term system user or simply users if we refer to both groups of users, collectively.

Five example scenarios of taxonomy exploitation from the business process “service booking” in a trade fair application are discussed here to illustrate the wide range of taxonomy applications in our framework. The examples are taken from the trade fair support system developed in the EU project FAIRWIS in the context of the ICKM meta-design framework at our department [22]. In “service booking” the fair exhibitors are the end users of the system, who select services like power supply from the service offer of the trade fair organizer.

1. In a service booking page a taxonomy is used by the trade fair exhibitors to navigate the service offering of the trade fair organizer and to select the service classes they want to book a service from. This is an example of taxonomy exploitation (*browsing*) in content management for the end user (see Fig. 2).

2. As a basis for service browsing the respective service taxonomy has to be defined and the service profiles of the offered services have to be classified into this service taxonomy subtree. This is an administrative task of the power user in the area of taxonomy and content management.
3. Considering the design process of the trade fair business Web application this booking page is itself a (design) object that can be classified. The page author, who acts as power user, may classify it, for example, according to the access rights using a user role as category and according to the language it is intended for. Furthermore the part of the taxonomy to be displayed has to be specified. This is done by defining a context of usage, which is described in more detail in section 4.3.
4. For other design objects the classification is just used internally. The set of available templates for the list view and the service profile view in the booking page offered to the power user during page setup is computed exploiting the adequate classification of these templates when they were designed.

Figure 2: An example "service booking " page

5. The taxonomy also provides a basis for matchmaking between information needs of content and knowledge consumers and available ICK resources, as it is used for qualification consulting, collaborative filtering, personalization, etc. In the booking scenario this can be exploited to propose adequate services to the exhibitors according to characteristics like stand size.

These examples motivate the following list of functionalities required for effective taxonomy meta-design support:

- User interfaces and interaction models for taxonomy browsing and content navigation;
- Taxonomy management for the construction and modification of taxonomies as well as for the classification of arbitrary information objects;
- Authoring tools for context definition and taxonomy page setup.

4.2. Classification Model Characteristics

Classification connects concepts which are part of the taxonomy with information objects representing entities from the user model, domain model, design model etc. (see section 3.1). In our approach multiple classification support, typed classification relationships, and functional taxonomy subtree refine the employed classification model and facilitates taxonomy context definition.

4.2.1 Typed Classification.

The classification model underlying the approach is extended by supporting different classification relationship types for connecting information objects to concepts instead of just considering the standard "is-a" relationship. This extended concept of classification enriches the resulting classification schemes by expressing more precisely the specific underlying classification semantic and enables specific processing based on the relationship types [15]. Coupled with the chosen multiple classification approach a rich class of statements can be made about an information object just relying on the classification concept. A user profile representing a fair organizer's employee may for example be classified as a service manager via a "is-a" link. Furthermore, his responsibility for a service family may be expressed via a classification of type "is responsible for" to the considered service family category (see Fig. 3) implying a notification upon service booking.

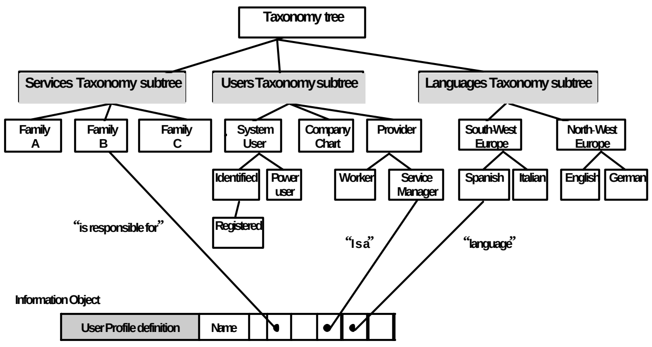

Figure 3: Multiple classification schema of a user profile "information object"

4.2.2 Functional Taxonomy Subtrees

In general, the concept contained in a taxonomy and their structuring depends upon the ICKM application under consideration. System functionality as well as the content area both influence the relevant concepts in a taxonomy-driven meta-design framework. Taxonomy definition, thus, is one of the tasks when setting up an ICKM system instance in the framework. When setting up a trade fair system, for example, the thematic area for the trade fair exhibitors have to be defined as a taxonomy subtree according to the requirements of the trade fair under consideration. However, taxonomy definition is not done from scratch; existing taxonomies may be reused and adapted. On the one hand, the taxonomy tree of similar applications like the taxonomy from the previous trade fair event may be reused. On the other hand, there are *functional taxonomy subtrees*, that fulfill a certain function in the ICKM meta-design framework and are used in a similar form in all or most ICKM systems. In our framework, we identified three important functional taxonomy subtrees:

- *User role subtree:* This subtree contains categories referring to the different user roles and serves the definition of the user model of the ICKM system. The set of supported roles may differ, but the subtree itself is required for supporting user model-based functionality like the setting up of the task model. Classification based on categories of the user role subtree is, for example, used for role-specific business process definition and for the definition of role-based access rights.
- *Language subtree*: This subtree contains the languages supported by the system. ICK resources are classified according to their language assignment. Note, that this is not necessarily the language used in the ICK resource. In absence of a German translation of some Web page, for example, the English page may be used instead. In the taxonomy itself multi-language support is handled by language-specific category labels.
- *Subject Subtrees*: These subtrees of the taxonomy cover the knowledge areas in the application domain of the ICKM system. Examples are the thematic area of the fair or the topics covered by the courses offered in an e-learning system. The subject subtrees play an important role for content navigation and management.

4.3. Taxonomy Reduction

As discussed in the previous section a taxonomy in our framework is involved in various tasks with respect to system design as well as with respect to the implemented ICKM business processes. This results in a considerable size of the taxonomy and the set of classified information objects. The underlying taxonomy thus should adapt to the current context of usage, reducing the taxonomy to the parts that are relevant in the current context. The goal of context-adaptive taxonomy reduction is to provide sufficient and at the same time only the relevant context to the classified information objects.

4.3.1 Taxonomies and their Context of Usage

Taxonomy reduction is based on modeling the current context of usage of a taxonomy. In our approach the following dimensions are taken into account in context modeling:

- Root of the relevant taxonomy subtree
- Taxonomy and Classification Language
- Accessing User Role
- Restrictive categories (set of categories that further restrict the considered classified resources)
- Allowed Resource types
- Resource set

In the "service booking" scenario from section 4.1, for example, a taxonomy context is defined for setting up the booking page. The author of the page chooses the language for the taxonomy, the root of the service subtree to be displayed and the accessing user role. The context dimensions are discussed in more detail when the reduction process is presented in the next section.

Taking a closer to the concept of taxonomy, it becomes obvious that taxonomies are not only embedded into the context of usage of the current business or design process. They also provide a context to the information objects they classify setting them into relationships to the concepts used for the classification and to other classified information objects.

4.3.2 The Process of Taxonomy Reduction

The process of taxonomy reduction, which adapts a taxonomy according to a given context, is inspired by the work in [16]. In this work taxonomies are used to systematically "summarize" the content of a resource collection by dynamically computing and pruning the taxonomy subtree defined by the categories associated with the resources in the collection. We generalize this approach by considering further taxonomy context dimension in addition to the underlying resource set.

All dimensions of the considered context influence the process of taxonomy reduction, which results in a reduced taxonomy tree and a set of classified information objects, the so-called *classification set*. The elements of the classification set are triples *(c,rt, io)* connecting a concept *c* of the taxonomy, a relationships type *rt* (used for classification), and the classified information object *io*. The following description summarize the algorithm underlying the taxonomy reduction process:

Subtree Root: The specified root *r* restricts the part of the taxonomy that is considered for reduction. Only concepts that are part of the subtree rooted in *r* are taken into account in the process.

Language: The language dimension influences the classification set as well as the category labels to be used in the taxonomy:

- For the classification set only those objects are chosen that are connected with the respective language node *l* in the taxonomy tree via a link of relationship type "language".

- Besides, for the taxonomy tree the labels of the chosen language *l* are returned. If no label for this language is available or no language *l* given the default language is used.

User Role: This dimension of the context is related to access rights. Only information objects connected with the specified user role via a "has access" link will be inserted into the resulting classification set. A missing access right specification for a design object is considered as "no access restriction".

Restrictive Categories: Only information objects classified under all of the given categories $c_1, .., c_n$ are considered for the classification set. This dimension enables the formulation of additional restrictions. The language and the role dimension of the context can be considered as special cases of restrictive categories that are related to a functional taxonomy subtree and require a specific relationship type.

Resource Types: The classification set is restricted to the specified types of objects by this dimension of the context. Only information objects which belong to one of the given resource types are considered for the classification set.

Resource Set: This represents an alternative but also complementing way of reducing the taxonomy. In this case the reduced taxonomy represents a systematic summary of the given resource set [16]. The resulting taxonomy consists of the union of the categories under which the objects in the resource sets are classified plus their ancestors in the taxonomy tree.

The reduction process starts with the consideration of the subtree *t* given by the chosen subtree root. To this taxonomy and the associated classification set the restrictions defined by the dimensions language, user role, restrictive categories, resource types are applied reducing the number of information objects in the classification set. The restrictions formulated by the different dimensions of the context are implicitly connected by a logical "and". This means, for example, that the objects in the classification set have to be of one of the given object types *and* they have to be accessible by the specified role.

After this reduction phase the taxonomy may contain concepts to which no object is associated in the classification set. An optional pruning phase that starts at the leaves the taxonomy recursively eliminates the concepts with no classified information objects connected in a bottom up fashion.

5. A Taxonomy-driven ICKM Meta-Design Framework

In the context of different projects at our department we developed an ICKM meta-design framework enabling the customized setup of ICKM systems. It is build up by an extensible set of application logic components from the area of content and knowledge management that can be flexibly composed into Web-based system instances. A suite of system authoring tools supports power users as well as developers in setup, customization, and evolution of such system instances. Using tools from this suite the user can customize the functionality and the user interfaces offered by the framework to the application domain and subsequently adapt the system instance to changing requirements of the underlying business process (the authoring tools suite is presented in more details in [17]). For the framework standard compliance and flexibility have been important design goals.

The taxonomy component is a core part of the framework architecture that is used in the authoring as well as in the operation of the system instance.

5.1. Framework Architecture Overview

In presenting the architecture we distinguish two environments: The authoring environment used for system setup, customization and evolution and the environment for operation of ICKM system instances.

5.1.1 Authoring Environment

In addition to task-specific authoring tools that are dedicated to the setting up of UI components in the context of a specific task, the extensible authoring tool suite of the framework also contains a set of authoring tools that support the definition of the Web application model in a more general way (see Fig. 4). The most important ones of these general tools are a publishing tool called "Form Manager", which enables the definition of form-based user interfaces for Web applications and their coupling with business data and application logic activities, and the data object manager, which enables the definition of a semantically enriched domain model based on the business data available in the application.

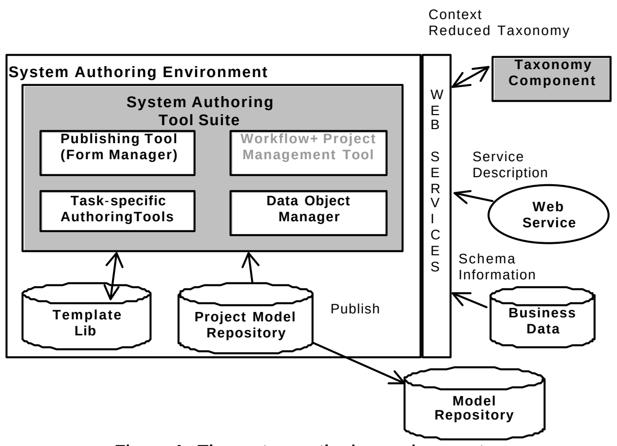

Figure 4: The system authoring environment

The design objects, object mappings, navigation maps, etc. created with the authoring tools are stored in a project model repository, where they are shared with other authoring tools and stakeholders involved in the system instance setup project. On completion of the Web application model for the system instance under consideration this model is published into the model repository, where it is available to control system operation.

Currently, an additional authoring tool supporting Web application setup projects and managing the underlying design and business workflows in a user-friendly way is under development.

5.1.2 Operative Environment

The operative runtime environment is responsible for running the ICKM system instance based on the Web application model constructed and customized with the authoring environment, the available business data and the functionality provided by the application logic components of the framework. This includes the following steps (see Fig. 5):

- For each business step the respective objects from the Web application model, like mappings, design objects, taxonomy contexts etc. are collected from the model repository and passed to the UI model compiler;
- Mappings defined during the authoring process are translated into bindings to business data instances and business application activities. This is done by the data object processor and the UI compiler . The activities are either standard operation like data storage or the sending of email or they are arbitrary application logic, all encapsulated as SOAP Web services [18].
- A processor translates the conceptual UI agent independent UI model into UI agent specific representations which can be displayed by the respective client. The approach used in the framework is based on XFORMS [19] the upcoming standard for form-based interfaces in the Web. The current processor translates XForms documents into XHTML representations (HTML + JavaScript).
- The Publishing Service manages the communication with the client and delivers the UI agent specific UI model.
- Following the paradigm of XForms much of the user interaction is handled on the client site. If client-server interaction becomes necessary, typically triggered by pressing a submit button, a Web service is activated referring to standard activities like data storage or other kinds of application logic. The underlying mapping has been specified as part of the Web application model.

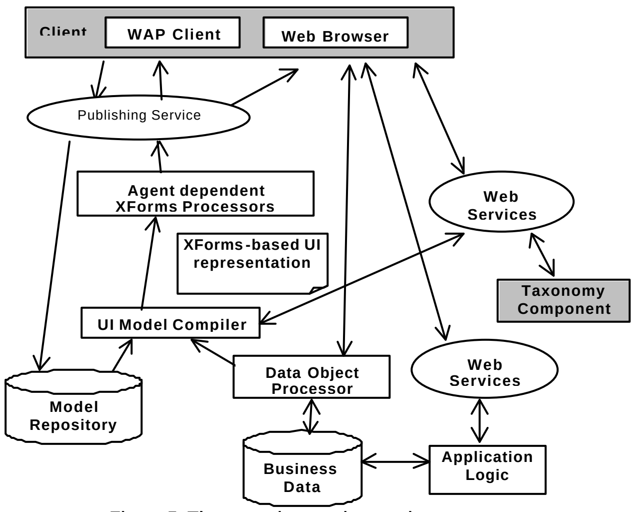

Figure 5: The operative runtime environment

5.2. The Taxonomy Component

The taxonomy component consists of four modules that implement the taxonomy support described in section 3 (see Fig. 6):

- The Taxonomy Manager provides core functionality for the construction and evolution of taxonomies like insertion and deletion of concepts, moving of taxonomy subtrees etc.
- The Classification Manager supports multiple classification of information objects with respect to the concepts of the taxonomy using typed relationships. As discussed in section 3 classification is not restricted to content objects but also applies to design object in our framework.

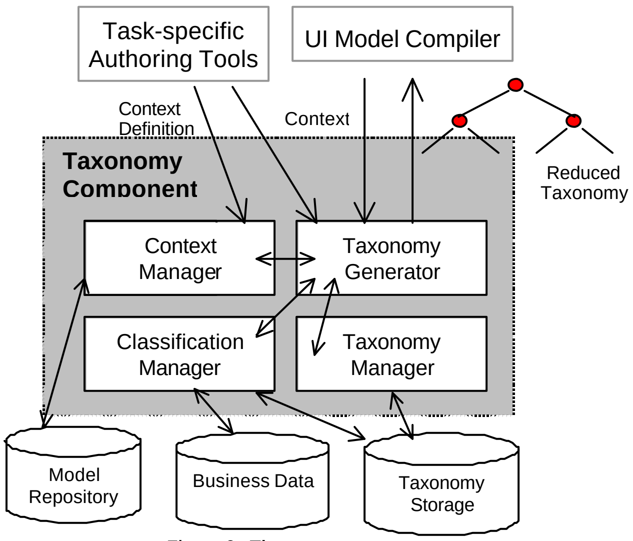

Figure 6: The taxonomy component

- The Context Manager enables the definition and storage of contexts for taxonomies. It is used in task-specific authoring tools like the booking manager to set up pages containing taxonomies for browsing or classification of information objects.
- The Taxonomy Generator implements the taxonomy reduction process. It receives a taxonomy and a context as an input and returns a reduced taxonomy as a result which is computed according to the approach described in section 4.3.

XML is used as an exchange format between the taxonomy component and the other tools and components, where the schema used for the description of taxonomies is based on RDF to ensure consistency with external components.

6. Conclusions and Future Work

In this paper we presented the taxonomy component of our ICKM meta-design framework. This component supports taxonomy management, information object classification and the set up of taxonomy pages in the design process for a Web application. It is used in system operation for navigating and organizing ICK collections as well as in system set up, customization, and evolution for navigating and organizing design objects and for the setting up of taxonomy pages. The support of context-adaptive taxonomies, which is realized by context-driven taxonomy reduction guarantees a focused, homogeneous view on context objects, design objects as well as business process activities. Furthermore, the use of a taxonomy as central organizational structure contributes to a common minimal ontological commitment between the different stakeholders involved in the Web application design process facilitating effective communication and cooperation as well as user empowerment.

The presented framework is still work in progress. Currently we are extending the framework by basic workflow management support that can be used for the modeling of the business process to be implemented by the Web application as well as in the design process for setting up a ICKM system instance. A further planned work package for the presented framework is its semantic enrichment by step-wise extension of the taxonomy component in the direction of an ontology-driven system [11]. Exploiting this enrichment flexible, knowledge-driven design support based on heuristics can be implemented.

As discussed in section 4 taxonomy construction is an iterative, evolutionary process where taxonomies are often not build from scratch but by adapting, refining and combining existing taxonomies. From the conceptual viewpoint the efforts needed for taxonomy construction can be reduced by systematically exploiting the wide variety of existing taxonomies and ontologies. From the technological viewpoint this raises the requirement for tools supporting the controlled integration of multiple taxonomies (taxonomy merging, see for example [20]). We are also planning to further investigate in these two aspects of effective taxonomy reuse.

7. References

[1] Dieter Fensel, Frank van Harmelen, and Ian Horrocks, and Deborah L. McGuiness, and Peter F. Patel-Schneider. "OIL: An Ontology Infrastructure for the Semantic Web". IEEE Intelligent Systems, 16(2):38-45, March April 2001.
[2] Gerald Fisher. "Social Creativity, Symmetry of Ignorance and Meta-Design", Knowledge-Based Systems Journal, 13(7-8): 527-537, 2000.
[3] P. Kruchten, "The Rational Unified Process: An Introduction", The Addison-Wesley Object Technology Series, 2000
[4] A Rational Software and Context Integration white paper, Building Web solution with the Rational Unified Process: Unifying the Creative Design Process and the Software Engineering process, http://www.rational.com
[5] B.A.Nardi, "A Small Matter of Programming", MIT Press, Cambridge, Mass., 1993
[6] T. R. Sumner, "Designers and their tools: Computer Support for Domain Construction", University of Colorado at Boulder, Ph.D. Dissertation, Dept. of Computer Science, 1995
[7] P. Pinheiro da Silva, "User Interface Declarative Models and Development Environments: A Survey", Proceedings of DSV-IS2000
[8] A. R. Puerta, "A Model-based Interface Development Environment", IEEE Software, pages 40--47, July/August 1997
[9] Donald A. Norman, "The Design of Everyday Things" – Originally published: The Psychology of Everyday Things – New York: Basic Books, 1988.
[10] Thomas R. Gruber, "A Translation Approach to Portable Ontology Specifications," Knowledge Acquisition, vol. 5, 1993, pp. 199–220.

[11] Nicola Guarino (ed.) "Formal Ontology in Information Systems", Amsterdam, Berlin, Oxford: IOS Press. Tokyo, Washington, DC: IOS Press (Frontiers in Artificial Intelligence and Applications), 1998.
[12] B. Chandrasekaran, R John. Josephson, V. Richard Benjamins, "What Are Ontologies, and Why Do We Need Them", IEEE Intelligent Systems 14(1):20--26, 1999
[13] Thomas R. Gruber, "Toward Principles for the Design of Ontologies Used for Knowledge Sharing" - In Formal Ontology in Conceptual Analysis and Knowledge Representation, edited by Nicola Guarino and Roberto Poli, Kluwer Academic Publishers, - Available as Technical Report KSL 93-04, Knowledge Systems Laboratory, Stanford University.
[15] Klaus Marius Hansen, Christian Yndigegn, and Kaj Grønbæk, "Dynamic Use of Digital Library Material - Supporting Users with Typed Links in Open Hypermedia", In Research and Advanced Technology for DigitalLibraries, Proceedings of the Third European Conference, ECDL'99, Paris, France, September 1999, LNCS 1696, Springer-Verlag, 1999
[16] Giovanni M. Sacco. "Dynamic Taxonomies: A Model for Large Information Bases", IEEE Transactions on Knowledge and Data Engineering, 12(3):468-479, May/June2000.
[17] Michael Fuchs, Claudio Muscogiuri, Claudia Niederée, and Matthias Hemmje, "An Open Framework for Integrated Qualification Management Portals", In Proceedings of The 3rd International Workshop on Management of Information on the Web (MIW'2002). September 2002, IEEE Computer Society Press, (to appear).
[18] Gudgin, Martin and Hadley, Marc and Moreau, Jean-Jacques and Frystyk Nielsen, Henrik, "SOAP Version 1.2 Part 1: Messaging Framework" - W3C Working Draft, 17 December 2001. http://www.w3.org/TR/soap12-part1/. December 2001.
[19] Micah Dubinko, Josef Dietl, Leigh L. Klotz, Roland Merrick, T. V. Raman, "XForms 1.0 - W3C Working Draft, 18 January 2002", January 2002.
[20] Florian Matthes, Claudia Niederée, and Ulrike Steffens, "C-Merge: A Tool for Policy-Based Merging of Resource Classifications", In Research and Advanced Technology for Digital Libraries, Proceedings of the 5th European Conference, ECDL2001, Darmstadt, Germany, September 2001.
[21] Muscogiuri, Claudio; Niederee, Claudia; Hemmje, Matthias; Fuchs, Michael (2001), "Towards Meta-Design for E-Business: Experiences & Challenges", In: Proceedings of the Human Computer Interaction Consortium Winter Workshops (HCIC 2002) January 30, 2002, February 3, 2002, Fraser, Colorado (USA)
[22] The FAIRWIS project- FAIRWIS Consortium. The EC Information Society Technologies Programme (IST), IST-1999-12641.
[23] Hallvard Trætteberg, "Modeling work: Workflow and Task modeling", in Computer-Aided Design of User Interfaces II :Proceedings of the 3rd International Conference on Computer-Aided Design of User Interfaces, Kluwer Academic Publishers, 1999
[24] Fischer G., "Domain-Oriented Design Environments", *Automated Software Engineering*, 1(2): 177-203, 1994