FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Competence Cockpit

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Magister der Sozial- und Wirtschaftswissenschaften

im Rahmen des Studiums

Informatikmanagement

eingereicht von

Johannes Daxböck
Matrikelnummer 0226015

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuung
Betreuer: ao. Univ. Prof. Dr. Jürgen Dorn

Wien, 02.05.2011

(Unterschrift Verfasser)

(Unterschrift Betreuer)

Erklärung zur Verfassung der Arbeit

Johannes Daxböck
Dornerplatz 10/6, A-1170 Wien

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit - einschließlich Tabellen, Karten und Abbildungen -, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

(Ort, Datum)	(Unterschrift Verfasser)

ABSTRACT

Today knowledge about competences plays an important role in individual and organizational efforts to achieve certain goals. Competence management has advanced from its origin within human resources and nowadays overlaps with related domains like knowledge management and learning management.

But often it does not take into account that everyday actions can provide valuable resources for competence management processes. These actions need to be formalized and made actionable through different measurement and aggregation methods. COMPETENCE COCKPIT, a framework to support different resources and methods for competence management processes is presented in this thesis.

The framework is based on the content management system Drupal and fulfills requirements for openness, customizability and modularity to ensure the carrying out of four core competence management processes: (1) competence identification, (2) competence measurement, (3) competence development and (4) competence usage. It can be used to build specific competence management systems for certain domains, allowing to focus their development on intrinsically domain specificities.

To achieve this, COMPETENCE COCKPIT can tap into different internal and external resources for either competence measurement, development or usage. These resources, managed by the underlying content management system, are used to build evidences, which subsequently can be used for competence profile calculations. The methods for competence measurement from different resources and profile calculation are exchangeable.

Further the framework allows concepts of already existing competence ontologies to be imported and reused. The customizability and modularity of COMPETENCE COCKPIT was evaluated by setting up an example system in the university domain, with domain specific resources and measurement methods as well as by implementing additional modules using the interface opportunities provided by the framework.

Finally ideas and approaches for additional enhancements and future development of COMPETENCE COCKPIT are discussed.

KURZFASSUNG

Heutzutage spielt Wissen über Kompetenzen eine wichtige Rolle diverse Potenziale effektiv zu nutzen. Kompetenzmanagement hat seinen Ursprung im Personalwesen und überschneidet sich inzwischen mit anderen verwandten Bereichen, wie Wissensmanagement und Lernmanagement.

Kompetenzmanagementsysteme ziehen jedoch oft nicht in Betracht, dass alltägliche Aktionen in Beruf und Freizeit als Ressourcen für Kompetenzmanagementprozesse herangezogen werden können. Diese Aktionen müssen für Kompetenzmanagement umsetzbar gemacht werden. Dies setzt verschiedene austauschbare Methoden zur Kompetenzmessung sowie zur Kompetenzprofilgenerierung voraus.

Diese Arbeit präsentiert ein modulares Framework für Kompetencemanagement, COMPETENCE COCKPIT, basierend auf dem Content-Management-System Drupal. Es unterstützt die Prozesse: (1) Kompetenzidentifizierung, (2) Kompetenzmessung, (3) Kompetenzentwicklung und (4) Kompetenznutzung. Mit Hilfe des Frameworks lassen sich spezifische Kompetenzmanagementsysteme entwickeln. Das Framework unterstützt dabei, den Fokus der Entwicklung auf bereichsspezifische Eigenheiten zu legen.

Um dies zu erreichen, erlaubt es verschiedene Ressourcen als Quelle für die Ermittlung und Messung von Kompetenzen zu nutzen. Die so erhobenen Daten tragen in weiterer Folge zur Berechnung von Kompetenzprofilen bei. Weitere Ressourcen in Form von Inhalten, welche über Drupal verwaltet werden, können ohne größeren Aufwand in das System integriert werden.

Die Methoden zur Kompetenzmessung und Kompetenzermittlung, sowie zur Kompetenzprofilgenerierung sind austauschbar. Die Modularität und Flexibilität von COMPETENCE COCKPIT wurde durch das Aufsetzen eines Beispielsystems im universitären Bereich evaluiert. Das Framework wurde um mehrere Module erweitert, welche die angebotenen Schnittstellen des Frameworks nutzen.

Abschließend bietet diese Arbeit Ideen und Ansätze für weitere Verbesserungen und Entwicklungen des Frameworks.

Contents

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Since its introduction, which is credited to McClelland (1973), the term competence and its use has evolved quickly. Competence-based approaches especially gained popularity and acceptance within the human resources community (Dubois, Shadden, Kaufman, & Brethower, 2000). They play an important role in organizational and individual efforts to ensure the achievement of strategic goals. Due to their growing use in different domains a multitude of definitions have advanced in the past. Table 1.1 presents a small fraction of them to provide a more complete understanding of what the term may incorporate.

The primary objective of competence management is the provision of information describing competences of different people or groups of people. It can be organized according to four main kinds of processes (Berio & Harzallah, 2005):

Competence identification: how to represent and design competences?
Competence measurement: when and how to identify and define competences acquired by actors and the relationship between actors and competences?
Competence development: when and how to acquire competences in a planned way?
Competence usage: how to use acquired and required competences?

The typical competence management system is an information system designed to support the competence management processes by allowing the storage, retrieval, identification and distribution of competence data. Also competence management systems are often integrated into existing organizational systems like knowledge management systems or learning management systems.

Especially knowledge management and competence management share a common trajectory (Baladi, 1999; Hong & Ståhle, 2005). In addition to assisting human resource development tasks, competence management systems nowadays also suggest instruments for resource allocation, knowledge management or e-learning support (Draganidis & Mentzas, 2006). In addition to that, social software features integrated into these systems leverage network effects to motivate the sharing of knowledge and competences (Jiang & He, 2007; Marenzi, Demidova, Nejdl, & Zerr, 2008; Dorn, Herzog, & Werthner, 2008). For instance Cohen and Prusak (2001) highlight the potential of highly networked employees to increase productivity and innovation.

In research and practice competence management started at a personal level, focusing entirely on the individual an its cognitive traits. However it has made a shift from focusing solely on the individual to also look at other levels (Lindgren, Henfridsson, & Schultze, 2004; Sanchez, 2003). Competence management gradually developed into also comprising some form of organizational level competence management, being either an aggregation of single competences across individuals or a concept of collaborative competence embedded in an organization. Terms like organizational-, group-, core- or strategic-competence came up (Lindgren et al., 2004; Nonaka & Takeuchi, 1995). The existence of such competence results from the understanding that the sum is bigger than the accumulating parts. The focus of such competence lies on shared problem solving ability, inter-personal competences in terms of fostering group atmosphere and handling of responsibilities. This also involves the active creation of competences and identification of future needs (Prahalad & Hamel, 2006; Scarbrough, 1998; Lahti, 1999). Recent research in competence development emphasizes on the role of inter-organizational relations and their implications. Concepts like virtual organizations, network competence or relationship competence evolved and attracted an increasing amount of attention (Sanchez, 2003; Kokko, Vartiainen, & Lönnblad, 2007).

For identification, representation and design of competences, a shared common model is needed. When talking about leveraging network effects, this model also needs to be sharable. Knowledge and competence ontologies provide solutions for sharing a common understanding within and between different entities. Ontologies can be thought of a machine-readable controlled vocabulary of concepts. For instance they can consist of a taxonomy[1] of competence term definitions with relations between these terms (Smith, 2003). Subsets of a such a competence term taxonomy can be attached to content like profiles (Dolog & Schäfer, 2005), learning objects (Knight, Gasevic, & Richards, 2005) or competence development programs (Woelk, 2002). Especially competence assignment that comprises a set of instances from the underlying ontology to user controlled profiles is increasingly im-

[1] I refer to an ontology as the specification of the characteristics of a domain and to a taxonomy as a hierarchical classification of entities within a domain.

portant. Since competences are often hierarchically structured, their representation with taxonomies seems suitable. Due to possible ontological relations between competences it may also be possible to infer additional knowledge.

Competences are acquired, developed or used all the time, through actions like the participation in courses, trainings, events, discussions or the creation and consumption of content resources. Competence management systems seldom take such actions into account for competence management processes. In today's world there is a vast amount of data on the web available, which can be thought of a formalization of such actions, from which knowledge about competences can be derived. Advancements in information extraction can make these knowledge resources actionable for competence management. For instance there is the possibility to tap into knowledge markets (e.g. Quora[2] or StackExchange[3]) for competence management processes. Also information and knowledge lying around in Wikis, Blogs or Microblog services can be valuable resources. The types of resources on the web where knowledge about competence data can be derived, seems to be without limits.

There exist some approaches in addition to manual assessment which try to derive competences from resources, like Q&A type resources (Dorn & Hochmeister, 2009) or publications (Rodrigues, Oliveira, & Souza, 2005). However these approaches often only focus on a single resource type or a specific measurement method. No holistic or abstract approach to allow the easy integration of different resource types and measurement methods seems to be available. There's also the problem that competence management systems seem to be closely tied to a specific group of users (e.g. HR-department) within an organization, which hinders the adoption of other user groups and of profile self management approaches (Lindgren, Stenmark, & Ljungberg, 2003). Users often have to deal with complicated data entry and updating methods (Stenmark & Lindgren, 2004). They seem to be not encouraged to enter new data, use idle resources or transform their actions into resources. Complicated and copious data input can be a major problem for system participation. Also the development of competence management systems is often too closely tied to a legacy system to which a competence management part was added on, which might have implications for possible extensions or customizations. Some systems prove to be quite inflexible in applying needed changes quickly in today's networked world (Harzallah, Berio, & Vernadat, 2005). Competence management has grown into a domain of its own, with own research being conducted upon. Without enough modularity the integration and adoption of new research methods into existing systems, which might be a competitive advantage, can be cumbersome.

Considering the evolvement of competence management system and looking at the requirements and problem fields identified above the following areas seem cru-

[2]http://www.quora.com/, 05.03.2011
[3]http://www.stackexchange.com/, 05.03.2011

cial for the development of a modern competence management framework:

- For competence identification, a competence model as foundation, representing the design of competences, needs to be provided. This model needs to be shareable across different domains and flexible enough to respect different characteristics of competences.
- Competence measurement, development and usage need to reflect that different internal and external resources can be made actionable for these processes.
- These resources can be very different in their nature. A one size fits all approach to competence measurement can't be utilized. An interface, being the least common denominator for assessment has to be developed. Implementations of this interface can then be applied to appropriate resource types. This may help the faster adoption of new research methods.
- Profile management is a crucial part of a competence management system. Users must be put at the helm of profile and identity management. Also profile management needs to be applicable on different levels (e.g. individual, groups, network,...). Competence data derived from different resource types needs to be aggregated and integrated into these competence profiles.
- A system must be flexible and modular, based on open standards to allow further extensions and the integration of outside services. Also it must leverage network effects to help encountering new resources and to give usage incentives.

This thesis focuses on the development of the main entities of such a framework, called COMPETENCE COCKPIT in the following, and its evaluation and takes the following approach:

1. Identification of a competence model to support core competence management processes, (1) competence identification, (2) competence measurement, (3) competence development and (4) competence usage.
2. Specifications of requirements for a modular architecture to bundle functionalities derived from this model.
3. Development of a set of modules, logically bundling the identified functionalities, which are based on the above requirements and specifications.
4. Evaluation of the framework through implementing an example system to showcase the framework's modularity, extendability and customizability.

Boyatzis (1982)	An underlaying characteristic of an employee (i.e., a motive, trait, skill, aspect of one‘s self-image, social role, or a body of knowledge) which results in effective and/or superior performance.

Zemke (1982)	Competency, competencies, competency models, and competency-based training are Humpty Dumpty words meaning only what the definer wants them to mean. The problem comes not from malice, stupidity or marketing avarice, but instead from some basic procedural and philosophical differences among those racing to define and develop the concept and to set the model for the way the rest of us will use competencies in our day-today efforts.
Scarbrough (1998)	Competencies are seen as objects neither owned by an individual or organization, but rather a continuous process of production and reproduction.
Lahti (1999)	A competency is a construct, because it is not directly measurable or observable. It really only can be proven to exist through indirect methods. The indirect methods are observing or measuring indicators of a competency (e.g. behaviors or attributes that can be readily assessed).
Jackson and Schuler (2003)	Skills, knowledge, abilities and other characteristics that someone needs to perform a job effectively.
Allen (2004)	Competency. A specific, identifiable, definable, and measurable knowledge, skill, ability and/or other deployment-related characteristic (e.g. attitude, behavior, physical ability) which a human resource might possess and which is necessary for, or material to, the performance of an activity within a specific business context.

Biesalski and Abecker (2005)	Competence is determined by the knowledge-based and network-driven ability of an agent and his environment to act alone or with partners, to satisfy indirectly or directly existing customer requirements optimally. By this sustainable added values are created in a competitive and superior manner.
Schmidt and Kunzmann (2006)	Competencies are defined as bundles of work-relevant skills, knowledge and abilities. Competencies are usually associated with competency levels to describe different degrees of an abstract competency type.
Draganidis and Mentzas (2006)	. . . competence seems to be typically defined in terms of, a *category*, a *competency* (descriptive name for a competency), a *definition* (basic explanation of the competency) and a demonstrated behavior, which demonstrates the possession of competency.
Dorn and Pichlmair (2007)	Competencies are described as abstractions of work-relevant human behavior that have emerged as a promising concept for making human skills, knowledge and abilities manageable and addressable.
Paquette (2007)	Competencies are statements that someone, and more generally some resource, can demonstrate the application of a generic skill to some knowledge, with a certain degree of performance.

Table 1.1: Competence definitions

The remainder of this thesis is of the following structure:

Chapter 2
explores literature in the field of competence management systems as well as systems of related domains like knowledge management and learning management. Special focus is given on competence measurement, application scenarios and the use of ontologies. Finally the Drupal content management system is explored together with some of its usage in research.

Chapter 3
defines the underlying competence model and describes it's entities. A set of modules and interfaces, connecting these modules, is extracted from the model. For these modules, requirements are specified and their core functions are described.

Chapter 4
presents the frameworks's architecture and design as well as the implementation of the framework's modules. Focus lies on re-using and adapting existing modules, structures and code facilitating the possibilities offered by the underlying content management system. Therefor I also describe the design and features of other Drupal modules, core or contributed, used for the framework.

Chapter 5
evaluates the competence model and the functionality implemented in the developed modules. A small competence management system for students is implemented using the framework. Different evidence types, manageable by users, are defined. Competence measurement can be done using these evidences as competence resource. Furthermore these evidences can be used to generate competence profiles, also manageable by users. Additional modules use interface opportunities provided by the framework. The aim of this system is to showcase the modularity, extendability and customizability of the framework.

Chapter 6
discusses shortcomings and implications of the implementation. It also features ideas and approaches for additional enhancements and further development.

CHAPTER 2

STATE OF THE ART

The following section explores literature in the field of competence management systems as well as systems of related domains like knowledge management system and learning management systems. Special focus is given on competence measurement, application scenarios and the use of ontologies. Finally the Drupal content management system is explored together with some of its usage in research.

2.1 Competence Management Systems

As already hinted in the introduction competence management shares a common trajectory with knowledge management. Competences of employees can be seen as important knowledge assets of an individual or organization. Hence competence management systems is often integrated into an organization's knowledge management. For instance to aid in finding appropriate individuals for tasks which require specific knowledge and competences. The same can be said for the learning management domain. Thus one can't look into recent scientific literature about one domain without glimpsing at the others. Furthermore I look at the use of ontologies as a solution to problems regarding system integration and understanding.

Hong and Ståhle (2005) identify perspectives on knowledge management and key conceptual views on organizational competence and its evolution. Their paper introduces an integrated and systemic view of knowledge and competence management based on the common trajectory both domains shared in recent times and their overlapping and compensative interplay. Also it highlights the challenge of managing such a system towards a self-renewable organization focusing on the task-specific functions in an organization. The managerial implication is that in today's business world, the key strategy is to facilitate self-renewal as a basic organizational capability.

Berio and Harzallah (2005) identify four main processes of competence management: competence identification, competence assessment, competence acquisition and competence usage. Furthermore they show, how knowledge engineering and related techniques like using ontologies, learning resources, expert rules or information retrieval can support these processes.

Hustad and Munkvold (2005) describe IT-supported competence management at the global telecommunications company Ericsson. The study highlights the potential role of IT systems for supporting strategic competence management. It also contributes to knowledge management processes in the form of knowledge networks and communities of knowing. The study investigates advantages in having global access to the company's competence resources in comparison to local competence management with regards to increase innovation and stimulate learning processes. However, it also illustrates how the realization of such a system is a challenging effort. This involves the specification and design of a competence catalog that includes competence levels covering both global and local needs as well as gaining acceptance and commitment from employees at various levels.

Lindgren and Stenmark (2002) present an 18-month action case study of the design, implementation and evaluation of a traditional competence system at Volvo Information Technology AB in Gothenburg, Sweden. As a result this study presents five design implications for competence management systems based on interest-activated technology:

- Action-based competence: search and locate people based on actions they perform (e.g. accessing, annotating, bookmarking, creating,...).
- Awareness of communities of interests: promote the establishment of informal networks and make individuals with similar interests aware of each other.
- Deeper level of personal information: include personal details about the organizational members and make this data accessible to everyone in the organization.
- Formal descriptions of competence: link dynamic information, personal details with formal descriptions of achievements and competences.
- Aggregation of competence data to visualize competence information known to the system.

Furthermore the study promotes a system with the potential to detect, visualize and leverage interests of organizational members. It shows how interest-activated technology can be exploited and developed to support competence management.

Lindgren et al. (2004) investigated competence management in six Swedish organizations over the period of 30 months. The research consisted of two action research cycles involving numerous data collection strategies, interventions and prototypes. The result is an integrative model of competence that outlines the interaction between organizational and individual level competence and the role of technology in this process. Also it incorporates a life-cycle typology of competence which differentiates between:

- Competence-in-stock; already gained but currently not used competence.
- Competence-in-use; currently used competence.
- Competence-in-the-making; competence which will be the result after a certain development process.

Liao, Hinkelmann, Abecker, and Sintek (1999) propose a competence knowledge base system (CKBS) which builds upon an ontology-based model of competence fields. Its use allows multi-criteria categorization, queries for personal competences, complex heuristic inferences for finding knowledgeable persons and easy integration of the CKBS into an overall organizational memory information system.

Draganidis and Mentzas (2006) find that open and semantic standards and self-service technologies are going to play an increasing role and continue to be integrated into competence management systems. They performed a feature examination of 22 competence management and 18 learning management systems.

Woelk (2002) describes a system for competence-based learning that uses ontologies and semantic web services to deliver learning objects to learners in a corporate environment. It highlights the use of web services based on SOAP, WSDL and UDDI that have been further augmented with semantic descriptions in an e-learning environment. This system is able to deliver personalized learning experiences that increase the effectiveness of corporate employees.

Knight et al. (2005) present an ontology based approach of integrating learning designs and learning objects. Goal is to facilitate reusability of learning designs with different content. A conceptual model is introduced as an intermediary between learning designs and learning objects. Ontologies facilitate the representation of these concepts. Finally the usefulness of the proposed approach is illustrated in three use cases: finding a teaching method based on domain-related competences, searching for learning designs based on domain-independent competences and creating user recommendations for both learning objects and learning designs.

Dorn and Pichlmair (2007) present a competence management system for universities. The system distinguishes competences in knowledge and experience aspects. Evidences, being learning objects like examinations, books read by the person, training, assessments, project work or e-learning courses are used to provide information about existence, sufficiency or level of a competence. They can be seen as explicit measurement of a competence and alter the characteristic of knowledge and experience aspects in a competence. The competence of a person is computed by the system at the point of time when it is required and all evidences until this moment are used for computation. An ontology provides a common vocabulary of terms for referencing competences and evidence types in the computer science domain. Competence profiles are presented in HR-XML format. A gap analysis algorithm determines differences between actual profiles and desired goal profiles used for further recommendation on courses for a student. Access to profiles can be enabled for e.g. recruitment to allow companies to find appropriate candidates for job vacancies. The data is stored in an encrypted XML data store.

Draganidis, Chamopoulou, and Mentzas (2008) integrate competence management, e-learning and an ontology in one prototype-system relying heavily on semantic technologies. They analyzed the system based on the usage scenarios for finding a best fit employee for a project and a future position skill gap report. RDF-cascading was identified as a main problem when using a system with deep semantic integration.

Berlanga, Bitter-Rijpkema, Brouns, Sloep, and Fetter (2009) claim that, in the context of learning networks, personal profiles are important to enhance social interaction and foster user contributions. They analyzed three different popular profile sites (Facebook[1], Myspace[2], LinkedIn[3]) in order to identify the information personal profiles should contain and to find out, how to motivate registration and how to stimulate contribution. Their results present people‘s perception regarding the information they consider important to get acquainted with members of a social network site as well as to present themselves to others in such a network. Results show that occupation, interest and expertise are considered important. Also, real name, school, employer and city are ranked in the top five positions. Number of contacts, hometown, and the communities the person is participating in, are not perceived as important.

Velardi, Cucchiarelli, and Petit (2007) describe a fully implemented, semantically indexed competence management system, used to facilitate research collaboration and coordination. The main highlighted advantages of ontologies are improved information access and interoperability. The paper gives experimental results to support these claims. They provide usage data on the competence management system and data to quantify the added value of semantic search.

In 2007 a competence ontology was introduced at Vienna University of Technology (VUT). This competence ontology has been extended and improved over time. The first iteration of the ontology (Dorn & Pichlmair, 2007) was based on a German language taxonomy featuring competences in the information systems domain. This taxonomy was the result of a student course at VUT. The next iteration featured an extension for concepts required in job metasearch applications (Dorn, Naz, &

[1]http://www.facebook.com/, 3.3.2011
[2]http://www.myspace.com/, 3.3.2011
[3]http://www.linkedin.com/, 3.3.2011

Pichlmair, 2007). This system provides solutions for automatic integration of data, structures and processes into a meta-search-engine with using the modeled domain ontology and multiple matchers. In 2009 drawbacks and desired extensions were identified through discussions with people working with and having interest in the ontology as well taking the needs of related projects like TechScreen (Dorn, Herzog, & Werthner, 2008) into account. The current version of the ontology can be found on the web[4].

Schmidt and Kunzmann (2006) present a reference ontology for human resource development, which brings together the disciplines competence management, knowledge management, business process management and technology enhanced workplace learning. In this conceptualization on the operational and technical level, technology enhanced workplace learning uses competences to foster learning activities of individual employees.

Trichet, Bourse, Leclere, and Morin (2004) present a system in which resumes and job descriptions are annotated with competences from an ontology, providing a common vocabulary for searching and matching functions in the human resource domain.

Braun, Kunzmann, and Schmidt (2010) argue that self-service and a bottom-up process in collaborative competence management provide more timeliness and usefulness in comparison to a strict top-down formal approach. They allow an informal ontology development process embedded into everyday work activities and focus on ease of use to motivate user contribution. People are tagged with competences, which then subsequently enter an ontology maturing process containing the steps collection, consolidation, formalization and axiomatization.

A survey regarding ontology visualization shows that ontologies are predominantly structured as hierarchies (Katifori, Halatsis, Lepouras, Vassilakis, & Giannopoulou, 2007). However, in many domains, ontologies tend to be quite large and complex, making them difficult to explore and present (Storey et al., 2001; Crowder et al., 2009). The visual information seeking mantra approaches the problem of representing large data with a basic principle, i.e., overview first, zoom and filter, then details on demand (Shneiderman, 1996). When dealing with large unknown data, the concept of information scents in the form of scented widgets (Willett, Heer, & Agrawala, 2007) can improve traditional user interface elements, providing users with more context and helping them in accomplishing tasks.

Breslin et al. (2007) present a conceptual framework for the reuse and interlinkage of existing, well-established vocabularies in the semantic web. They present relevant properties of the Friend of a Friend[5] (FOAF) ontology usable for matching people and their skills in social networks. Then they detail the Semantically-Interlinked

[4]http://www.competencies.at/Ontologies/Competence/Competence.owl
[5]http://www.foaf-project.org/, 3.3.2011

Online Communities[6] (SIOC) project and methods for identifying relevant discussion topics or individuals. Finally they outline a scenario, combining FOAF, SIOC and topic classification using Simple Knowledge Organization System[7] (SKOS) that allows to find experts in a domain of interest. In contrast to this rather broad ontologies, there also exist initiatives, such as the IEEE Reusable Competency Definition[8] or HR-XML (Allen, 2004), which try to define broader domain models and schemas. But they can only be the least common denominator and therefor may lack important information in terms of context or expressiveness needed for the description of complex competence profiles and requirements (De Coi et al., 2007).

2.2 Competence Measurement and Usage

Among the most difficult problems within competence management is the problem of competence measurement. A measurement must provide a valid comparable representation of competences of an actor. When dealt with it in a manual way, which means that a human actor takes out the measurement process, we can distinguish between three approaches (Dorn, 2010):

Expert Assessment an assessment by an expert. This type can be tied to certain conditions (e.g. the competence of the expert is better than those of the probands).

Peer Assessment an assessment by peers. This type also can be tied to certain conditions (e.g. peers are of the same competence level as the proband).

Self-Assessment an assessment by the proband itself.

Further investigation must be done whether to use a qualitative approach or a quantitative approach for measuring competences (Erpenbeck & Rosenstiel, 2003).

As hinted in the introduction, the web provides an immense amount of knowledge resources which could be used competence management processes. Therefor advances made in the domains of text-mining, semantic web and information extraction, are gaining popularity within the competence management domain. Also they are often directly injected into the competence management usage process for instance for expert finding.

Becerra-Fernandez (2006) presents the role of ontologies and web mining techniques in the construction and maintenance of expert profiles. Her article also discusses the implementation details of two expertise-locator knowledge management systems.

Farrell, Lau, Nusser, Wilcox, and Muller (2007) present a system, which allows users to tag themselves and others with keywords. They show that people-tagging

[6] http://sioc-project.org/, 3.3.2011
[7] http://www.w3.org/2004/02/skos/, 3.3.2011
[8] http://ltsc.ieee.org/wg20/, 3.3.2011

has the valuable effect of collectively maintaining each other's interest and expertise profiles. The authors also conducted a study showing that most of used tags provide an accurate description of a user's expertise and interest. Moreover no offensive, abusive or inappropriate use of the system was discovered. They also used tags as simple voting system. The more people tag one user with the same tag, the higher the ranking of this user is, for this specific tag.

(Rodrigues et al., 2005) created a system, which mines researcher's publications for important terms by traditional text-mining methods. The relation between terms and a researcher's competences, derived from these terms is inputed manually by a person responsible for that task.

Oliveira et al. (2006) created a scientific knowledge management environment in which researchers may share data, experiences, ideas, process definitions and obtain all the necessary information to execute their daily tasks. One service of the system is responsible for user profiling and knowledge matching. It tries to identify researchers' interests, profiles and competences. This service provides information such as searching for users with similar profiles, who might be interesting to contact, discovering researchers' competences, suggesting experts to execute a certain activity and representing researchers' interests. The system distinguishes competences according to four categories:

- Declared competences: competences which a researcher declares.
- Project competences: correspond to competences which were the pre-requisite of a project.
- Extracted competences: recovered from a researcher's publications using text mining methods.
- Community competences: collected from the communities in which a researcher participates or contributes to.

Ehrlich, Lin, and Griffiths-Fisher (2007) find that current systems for finding experts do not adequately address social implications with regards to the search process. Their paper provides a user study of a social-context-aware expertise search system that can be used to identify experts, see dynamic profile information and get information about the social distance to an expert. This information can be helpful in the decision whether and how to initiate contact. The system also infers content and dynamic social networks from email and chat logs.

The university competence management system described by Dorn and Pichlmair (2007) is also used by Dorn, Pichlmair, Schimper, and Tellioğlu (2008). They describe experiences made in a software project management course, capturing and developing students' competences relevant in this area. An approach also applicable in commercial software projects. An experiment in measuring competences and competence enhancement is presented and evaluated. Assessments were done by self-assessment, peer-group assessment and assessment by a supervisor and implemented by structured questionnaires.

One usage scenario of TechScreen (Dorn, Herzog, & Werthner, 2008) was implemented and examined by Dorn and Hochmeister (2009). They describe an approach to mine for competences of users in a social software system. The system was implemented using Drupal, OWL and the Yahoo text mining service. The experiments were executed by university staff and students, the domain limited to Information Technology. Assumption is that sharing knowledge about a certain domain might be evidence for a certain degree of competence in this area. Competence calculation is done by collecting all content of a user, creating term lists, looking up the terms in an ontology and assigning them to a specific competence area. Using this methodology, competence profiles, represented in a taxonomic structure were generated and compared with the self-assessment of participating users. Building upon this scenario Daxböck and Hochmeister (2011) present an integrated user interface that supports users during competence self-assessment and that facilitates a clear presentation of a competence profile. User interface elements for competence ontology navigation, competence assessment and competence profile representation were introduced.

Colucci, Di Noia, Di Sciascio, Donini, and Ragone (2007) propose an ontology based system facilitating knowledge elicitation and extraction of core competencies automatically on company level. The system deals with the automatic clustering of companies in knowledge classes according to the competence they hold. They show that a company may extract its core competences and locate itself in a clustered knowledge network. The extracted competences may belong to different knowledge fields. The automation of competence extraction instead of manual annotation also allows the discovery of neglected or hidden skills relative to different branches.

Sieg, Mobasher, and Burke (2007) present an approach to personalized search that involves building models of user context as ontological profiles by assigning implicitly derived interest scores to existing ontology concepts. It uses a spreading activation algorithm to maintain the interest scores based on the user's ongoing behavior. The experiments in this paper show that re-ranking results based on the interest scores and the semantic evidence in an ontological user profile is effective for obtaining the most relevant results.

2.3 Drupal

The following section explores and describes the current state[9] of the Drupal content management system as well as its application in the scientific field.

[9] *By the time of publication of this thesis a new major version of Drupal will be available rendering some information in this section out-of-date.*

Drupal is a highly modular, extensible, standards-compliant, open source web content management framework, written in PHP. It's design separates cleanly between content management and content presentation. It can be used on different combinations of operating systems, relational databases and servers.

Drupal's core is a lightweight framework responsible for handling content-, user- and session-management, url-aliasing, localization, templating, syndication and logging along with a set of common library functions. It also features a low-level database API providing security and code portability across different databases. The core acts as a glue layer to stick together modules and is designed to provide a uniform method of extending Drupal. Modules can be woven together to create a robust feature set without copious amounts of code. Figure 2.1 provides a graphical overview of the Drupal architecture. A Drupal site can consist of three kinds of modules:

Core modules : ship with Drupal and are approved by the core developers.
Contributed modules : were written and shared by the Drupal community.
Custom modules : were created by the developer of a specific website.

Drupal makes use of the inversion of control design pattern, in which module functionality is registered and called at the appropriate time. These opportunities for modules to run their code are called hooks, which can be thought of as internal Drupal events. Hooks are implemented by functions prefixing a unique hook name with the module name implementing the hook. Therefor more than one module can implement a hook. Drupal examines all of the currently enabled modules, looking for functions that follow specific, predefined patterns. When it finds such functions it executes them, and uses the data these functions return to build a response. Hooks can be invoked directly in a specific module, which returns a single value, or all hooks of a certain name can be invoked in all modules implementing this hook, which returns an array of return values.

Nodes

A node can be thought of as an arbitrary piece of content (other main pieces in Drupal are blocks, comments and users). The module responsible for node handling is a core module in Drupal. By default, a node simply holds a title and an optional body field. Nodes are the main units of information available for display in Drupal. A node can be thought of as a generic data object stored within the Drupal managed database. For this object Drupal provides a consistent API for access, expansion and storage. A node also contains a basic set of behavioral properties (e.g. a node-ID, workflow status, title,...) and a set of local action tasks (e.g. edit, view, publish,...). Drupal separates different types of nodes into content types and allows adding different content types as well as role-based access management based for these types.

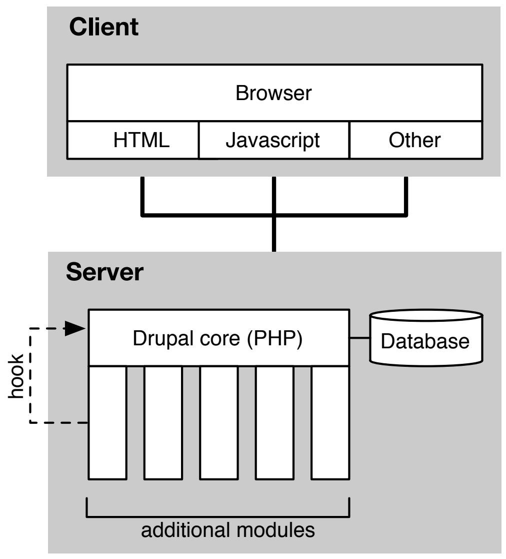


Figure 2.1: Drupal architecture

Content Construction Kit Content Construction Kit (CCK) is a group of modules that allows the definition of different content types for the creation of a site's content nodes. Without these provided mechanisms, one would have to store all data in a content type's body property. CKK assists in the creation of new fields for content types enabling a user to easily draw down a creation form, placing all of the content type's fields in one place. Fields consist of a field type and a widget.

The base install of CCK comes with a number of field types (text, integer, node-reference,...). Field types are not restricted to a single piece of data representation. For instance, one could collect text data through a text-field, text-area, a drop-down-menu, etc. Also, once you have defined a field type for a specific content type, CCK makes it easy to simply add that field type to different content types. Field can be encapsulated in their own modules. There exists a recommended path of implementing additional CCK fields in Drupal[10,11]:

1. Implement the CCK field in a Drupal module. Use *(module_name).info* file for general information, *(module_name).install* file for creating and removing the field's database schema and system variables and for registering the field with CCK and Drupal. The remaining functionality goes into the *(module_name).module* file.

[10] http://www.poplarware.com/articles/cck_field_module/, 3.3.2011
[11] http://www.lullabot.com/articles/creating-custom-cck-fields/, 3.3.2011

2. Define the field by implementing some Drupal core and CCK hooks that tell CCK about the field. These hooks are responsible for storage of field data, the field settings form, the definition of default values and the management of empty values.
3. Define one or more widgets. A widget is the name CKK uses for a form, used to edit the field.
4. Define one or more formatters with accompanying theme functions for the field. These are responsible for the actual data rendering.

CCK further provides greater control over the storage and theming mechanism. It creates a database schema that can easily be used by other modules allowing richer integration and further separation of content from presentation by allowing different visualizations through widgets for the same content field. A widget is a special piece of field data that is dependent on the field type. Widgets can be of different shapes (e.g. drop-down-lists, autocompletion-text-fields, tables,...). Breaking down a node's content into pieces of fields also promotes performance through better caching and better data integrity. CCK's basic feature set not only greatly increases the flexibility of Drupal's standard node structure. It also features an API adding the ability to create new fields or field representations (aka widgets) programmatically by introducing new modules.

Taxonomy

Drupal also comes with a taxonomy module that classifies content. Vocabularies consist of a collection of terms. A term is a label that will be applied to a node. Drupal adds a level of abstraction to all terms that are entered and refers to them internally by a numeric system-wide unique ID, not by name. Also synonyms, related terms and hierarchical relationships through an adjacency list model are supported.

Users, Roles and Permissions

User records are maintained by a user object type. User data is stored in a database and drawn out during processing. Information about a user is used for purposes such as authentication, determining preferences and permissions, logging and defining the ownership of nodes. Permissions are closely linked to the user object. Drupal provides a role-based mechanism for granting permissions to collections of users. In a nutshell, a user belongs to a role and permissions are granted to or revoked from a role. Using additional modules, permissions can be tweaked down to the level of tasks allowed on a node's content field for a role, which further enhances the granularity of the permissions system.

Drupal in Research

Corlosquet, Delbru, Clark, Polleres, and Decker (2009) present a system based on Drupal that enables site administrators to export their site content model and data to the Web of Data without requiring extensive knowledge on semantic web technologies. The modules present a number of extensions to Drupal that enable the exposure of site content as Linked Data and likewise allow the aggregation and reuse of existing RDF data from the Web:

- The RDF/CCK module auto-exports the content model of a Drupal site to an ontology and enables the exposure of a Drupal site's content as RDFa annotations.
- The EVOC module links to existing properties and classes from other semantic web vocabularies by subclass/sub-property relations and offers search functionality for commonly used vocabulary terms.
- The RDF/SPARQL Endpoint module exposes a SPARQL endpoint on the site data without additional configuration steps.
- The RDF/SPARQL Proxy module allows to dynamically load data into the site and displays this data using a lazy loading strategy for minimizing delays in the user experience.

All four modules offer possibilities to create networked web applications and push forward further population of the Web of Data by significantly lowering entry barriers for a larger user community.

Peacock (2009) makes the case that Drupal is ideally equipped to be used as a base system for creating a custom social networking site similar to the likes of Facebook or MySpace, especially for niche social networking web sites that can help to promote businesses, products, projects, and hobbies of any nature. The book begins with the fundamental concepts of a Social Networking site, and how Drupal can be used to create such a site. It then goes through installing and expanding the out-of-the-box Drupal feature set with third-party modules designed for Social Networking. Finally it looks at security, deploying, execution and maintenance of the social networks implemented with Drupal.

Drupal also builds the basis of the TechScreen usage scenario described by Dorn and Hochmeister (2009) as well as the proposed user interface for competence self-assessment presented by Daxböck and Hochmeister (2011). An example web search will reveal that Drupal, because of its flexibility, is used for a lot of other purposes in wide range of domains including knowledge-, learning- and competence management.

Chapter 3

Requirements

This section defines the underlying competence model of COMPETENCE COCKPIT and describes its entities. A set of modules, logically bundling functionalities derived from the model, and interfaces, connecting these modules, is extracted from the model. For these modules, requirements are specified and their core functions are described. Also requirements for the foundation of the framework are presented.

3.1 Competence Model

The following section describes the entities of the competence model used for COMPETENCE COCKPIT. They are based on the ideas presented by Dorn and Pichlmair (2007); Dorn et al. (2007); Dorn and Hochmeister (2009). These entities are used to carry out the processes of competence management: (1) competence identification, (2) competence measurement, (3) competence development and (4) competence usage. Evidences and competence profiles are the main resources to accomplish these processes.

Actor
: is a uniquely identifiable user in the system (e.g. an individual, a group of people, a whole organization,. . .)

Competence
: is the capability of an actor required to perform certain professional activities with a degree of performance. A competence is defined by a name and can be

measured for certain characteristics (e.g. theoretical knowledge, practical experience,...). In the following this characteristics will be referred to as competence values. These competence values lie within a normalized range describing percentages between 0 and 100. Ranges can be partitioned and mapped to qualitative scales of mutually exclusive levels (e.g. Basic/Advanced/Expert, Elementary/Intermediate/Advanced,...). A Competence can have parental relations to other competences, forming a competence taxonomy ranging from specific to more general competence concepts.

Evidence

is a measurement for the characteristics of a set of competences. They are used to capture information to prove the existence, sufficiency, level or possible development of a set of competences. They are based on arbitrary content types (e.g. reports, test results, evaluations, certifications,...), which are annotated with a set of competences and their corresponding characteristics. In the following these types will be called evidence types. The measurement can be of different methods (e.g. self-assessment, peer-assessment, automatic competence mining,...), which are tailored to the underlying content type. A measurement can be triggered by an actor. Every evidence is linked to an actor, to whose competence profile the evidence contributes.

Competence Profile

a profile is a structured representation of competences together with their respective values linked to an actor. The structured representation is derived from an aggregation considering all evidences linked to this actor for competence calculation. This aggregation is based on a specific and exchangeable algorithm. Furthermore we can distinguish between acquired and required competence profiles, which serve different purposes:

Acquired Profiles

specify accomplishments (in terms of competences) of an actor. These are typically used in order to show or prove that competences have been acquired.

Required Profiles

specify requirements (in terms of competences) to be fulfilled. These can be used for job descriptions, program prerequisites or team-building needs.

Figure 3.1 provides a graphical overview of the entities and their relationships.

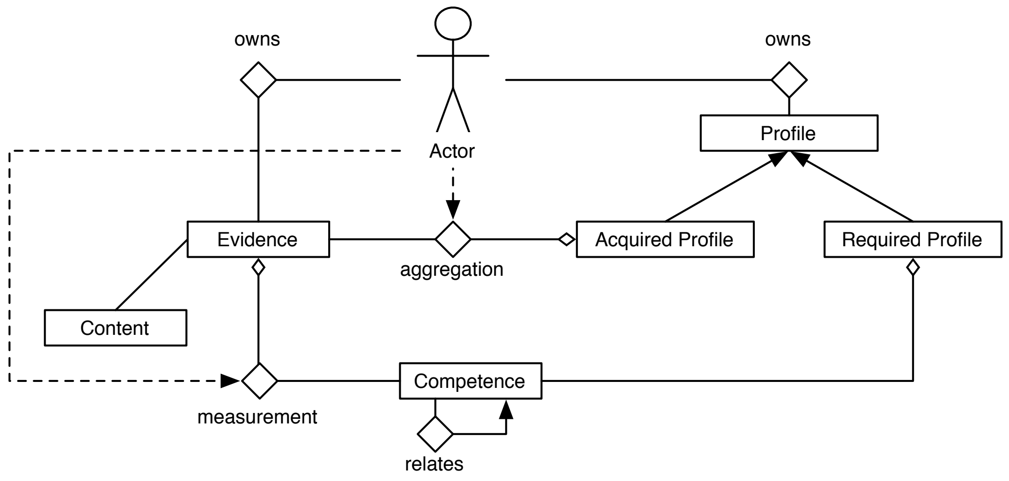


Figure 3.1: Competence model

3.2 System Requirements

Different methods for competence measurement, mining and competence profile aggregation, respecting different characteristics of competences have to be provided. Data, in the form of evidences and profiles, is used as resources for competence management. This data has to be stored and managed by an underlying system. A clear separation between data and data representation is needed. Also the system must be able to link data to existing ontology entities to support semantic integration. This integration shall foster comparability and context awareness.

Roles and permissions must provide fine-grained access control. Questions, who uses what parts of the system in which way, have to be answered, so that users are motivated in using the system without compromising privacy and protection to advocate trust and strengthen data validity. Such a permission model must be able to be tuned down to the smallest entities for every action to every building part of the system. The user interface needs to be unobtrusive, clear, responsive, functional and task based in order to allow easy and fast maintenance of data.

In order to decrease learning and development efforts put into the framework, it should be based on an already existing system, which fulfills the above stated requirements

3.3 Module Requirements

This section bundles functional requirements derived from the competence model (Figure 3.1) for different areas. The functions are used to support the core processes of competence management: (1) competence identification, (2) competence measurement, (3) competence development and (4) competence usage. The requirements are bundled in the following way:

Storage
: requirements regarding the underlying storage mechanism using a relational database.

Ontology
: requirements for managing and importing a competence ontology.

Competence measurement
: requirements for the interface, responsible for annotating content with competence data to build evidences. The interface will be called *Competence Broker* in the following.

Competence aggregation
: requirements for the interface which is responsible for the aggregation of competence data based on evidences to generate an acquired competence profile. This interface will be called *Competence Aggregator* in the following.

Evidences
: requirements to allow the administration and management of evidences.

Competence Profiles
: competence profile management is a crucial part of an actor's identity management. This section defines requirements allowing a maximum of control over the content of an actor's profile.

COMPETENCE COCKPIT deals with competence management on an individual level. Therefor in the following the term user will be used instead of actor. Regarding different types of actors, the system however should be designed to be customizable enough to also provide competence management processes on other levels (e.g. group, network, organizational,...). A user performing an action below is referred to as a user of a certain role containing the permissions to perform this action.

3.4 Storage

The database design of the underlying foundation has to be reused as widely as possible. It has to able to store information about content, content metadata, users, roles, permissions and action logging. The system must provide separate tables for storing competence data. Current and historical data must be distinguishable. Further separation must be done between current competence profile data and evidence competence data. The system has to provide a simple API for database operations to allow the development of add-ons. This for instance allows tracking changes in competence data on the database level for analysis purposes.

Ontology

The framework has to provide management and import functionality for competence ontology concepts. Different file formats for the ontology need to be supported (e.g. Turtle, N3, RDF/XML,...). The location of the ontology could either be the web or a local file uploaded to the server. Also the entry point in the ontology needs to specify-able. The module responsible for importing must be easily extendable through sub-modules responsible for specific concept imports (e.g. competence term taxonomy, value partitions, competence characteristics,...). As a minimal requirement a competence term taxonomy needs to be imported. Vocabulary management tasks, like adding, updating, deleting terms or defining term taxonomies have to be provided. No syncing from the system to the ontology will be done. No update, delete or 2-way-sync functionality will be implemented, since referentially triggered action clauses may pose problems and lead to unpredictable results. Other ontology management tasks are completely separated from the system and shall be done by applications designed for such a task (e.g. Protégé[1]).

Competence Measurement

The main function of a *Competence Broker* is the provision of an interface between content and a competence term taxonomy. The interface has to be designed to work with arbitrary resource types, so that they can access all content available for identifying competences to build an evidence. Within a *Competence Broker* the competence term taxonomy as well as the competence characteristics this broker uses have to be customizable and default to global variables. A user interface, which hooks itself into an evidence's edit form allows to access the broker's functions. An example implementation of this interface, providing a user interface for adding competences through assessment has to be given. This user interface also needs to take the problem of taxonomy navigation and value assignment into account. Since assessments, in the form of self-, peer- or expert assessments are a widely used approach for competence measurement, next to automatic competence mining, an unobtrusive, fast and accurate assessment process is of special importance.

Competence Aggregation

Competence Aggregator provides an interface between evidences and competence profiles. It uses the competence data provided by evidences to generates a competence profile. Within an aggregator the competence term taxonomy as well as the competence characteristics the aggregator uses have to be customizable and default

[1]http://protege.stanford.edu/, 05.03.2011

to global variables. The implementation must allow easy recalculation. The user interface, which triggers the aggregation algorithm has to hook itself into the edit form of a competence profile content type. An example implementation of this interface, providing a simple aggregation algorithm has to be given.

3.5 Evidences

Users must be able to define different evidence types. These types may contain arbitrary data as well as metadata fields like (e.g. tags, likes, ratings,...). At least one data field must contain competence data and another field must reference the competence profile an evidence contributes to. It must be choosable, whether an evidence will be used for calculation in a competence profile or not. This allows to manage evidences for other users too.

Within the definition of the competence data field the competence term taxonomy and the *Competence Broker* have to be customizable and default to global variables. Moreover the representation of competence data must be exchangeable. Default is a tabular style representation with basic sorting and filter functionality. Users must be able to update the competence data by calling the defined *Competence Broker* on demand.

Furthermore every evidence type must be manageable by users through create, update and delete operations. The number of instances of a specific evidence type for a user can be limited. Evidences can be made accessible to authorized users. Search and filter techniques shall allow a sufficient level of search and retrieval functionality for evidences within the system.

3.6 Competence Profiles

Strong identity and profile management including personal competence profile management is one of central functions of COMPETENCE COCKPIT. System use must be motivated by easy access methods, less restrictions and ease of use, to provide as much information as they want. Privacy concerns have to be taken into account by providing a fine grained role based access system.

Users must be able to define a competence profile. This profile has to include at least a data field with competence profile data. Within the definition of this field the competence term taxonomy and the *Competence Aggregator* have to be customizable and default to global variables. Moreover the data representation of the competence profile must be exchangeable (e.g. profile diagrams, radar charts, iconic representations, knowledge maps, mind maps,...) and made exportable to different output formats (e.g. JSON,HR-XML,...). Default is a tabular style profile representation with basic sorting and filter functionality. Also a set of links referencing initial evidence creation forms must be definable. This should give the user an incentive to create a profile and an initial number of evidences on account creation.

Users must be able to update their competence profile on demand. The updated profile must reflect all underlying changes in a user's evidences according to the *Competence Aggregator* used. The competence profile representation must be sortable and contain filter elements as well as provide links to the evidences, which contribute to a competence's values.

Users must also be able to distinguish between required profiles and acquired profiles. Competence profiles can be made accessible to authorized users. Additionally profiles can be expanded by arbitrary data fields to provide a more detailed description. Search and filter techniques shall allow a sufficient level of search and retrieval functionality for competence profile data (e.g. for expert finding).

Roles and Permissions

All functionality described above has to abide by a fine-grained system of roles and permissions. Users hold roles. Roles contain permissions which correspond to actions users are allowed to or omitted to perform. Permissions are defined for every action in the system on a user, content and field level. Every additional module, which performs actions not corresponding to these levels must implement additional permissions accordingly. A set of predefined roles showcasing different permissions on different levels has to be implemented.

Chapter 4

Design

This section presents the framework's architecture and design as well as the implementation of the COMPETENCE COCKPIT[1] modules. Also it describes the underlying Drupal system and its use within the framework.

4.1 Architecture

A web content management system, Drupal[2], was chosen as foundation for development. Content management systems usually provide an extendable architecture. Modules can connected by using interfaces and APIs. Content management systems, as their names suggests, aid in flexible management of content. They are designed to allow a large number of people to contribute and share data and allow granular access control to content. Also they provide a clear distinction between data and data representation as well as incorporate semantic- and social software technologies in an easy way.

Drupal was chosen because of its modularity and sophisticated content and content type management. Content derived from internal or external resources must be used for competence management processes. Such content is seen as evidence for a certain set of competences an actor possesses or develops. These evidences are subsequently used to generate competence profiles, which again can be seen as man-

[1]Module source code, distributed under the MIT license, can be found at `github.com/johdax/competence_cockpit`

[2]`http://www.drupal.org/`, 05.03.2011

ageable content. Users are put in control of evidence content, competence profile content and the methods for measurement and aggregation,

Competence data, needed for measurement, comes from the competence term taxonomy of a competence ontology. Drupal also features mechanism for managing lightweight taxonomies. Figure 4.1 shows the framework's architecture. As already mentioned, to reuse as much functionality from Drupal as possible, evidence and competence profiles are based on Drupal content types. What distinguishes them from other content types is a special data field for competence data in a content type representing evidences and a special data field for competence profile data in a content type representing competence profiles. To allow as much flexibility as possible and adhere to the Drupal way of assuring modularity these fields are implemented in their own dedicated CCK based modules *Competences* and *Competence Profile*.

Since *Competence Broker* and *Competence Aggregator* are only interfaces responsible for competence measurement and competence aggregation I provide a sample implementation for each interface. One *Competence Broker* implementation is a user interface, which can be used for self-, peer- or expert-assessment. The provided *Competence Aggregator* implementation is a simple algorithm for competence aggregation through the calculation of the mean values of each competence value annotated to published evidences.

The *Ontology Importer* is a module responsible for importing concepts from an existing competence ontology into Drupal. It's most important task is the import of a competence term taxonomy into the Drupal taxonomy system through its own submodule.

As mentioned above COMPETENCE COCKPIT consists of a set of custom modules making use of functionality provided by Drupal core and especially by the CKK set of modules. I make heavy use of the CCK API to define two special fields for storing competence data and competence profile data. Related to these two modules is a set of accompanying modules responsible for competence mining, competence assessment, competence profile aggregation and ontology mapping.

4.2 Ontology Importer

This module provides one-way mapping from an ontology to Drupal data structures. It uses ARC[3], a flexible RDF system for the semantic web written in PHP. Main function of the module is to parse an ontology in different formats (N-Triples, RDF/XML, Turtle) from a definable point of entry. These concepts will then be imported into Drupal. Ontologies can be stored in a local or remote file system. The module features its own hook system and hooks itself into the Drupal site configuration.

[3]`http://arc.semsol.org/`, 05.03.2011

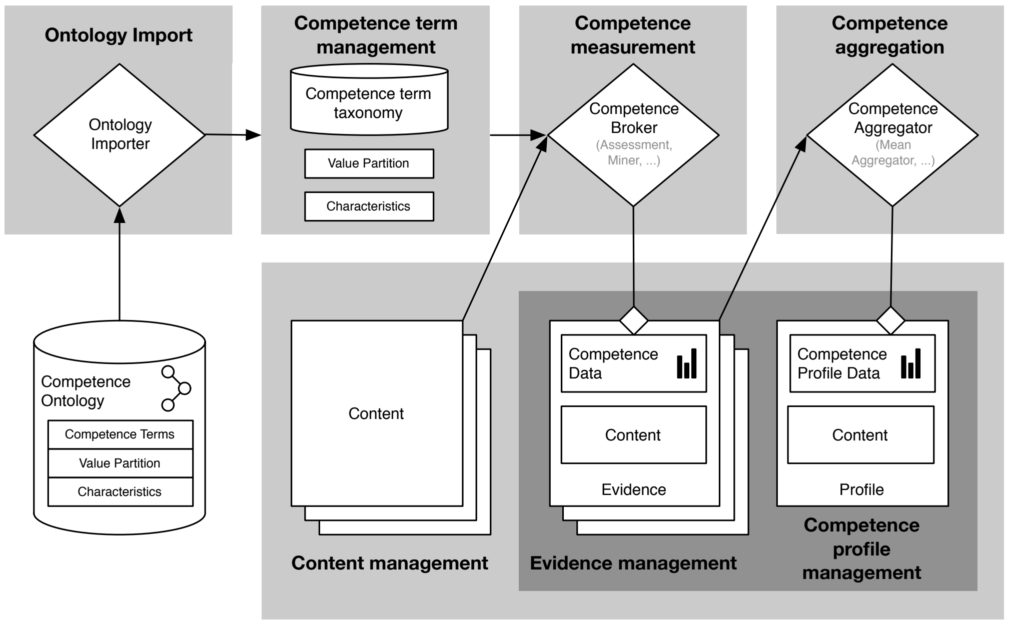


Figure 4.1: System architecture

Sub-modules implement the specific ontology entity import. A sub-module achieves this by implementing the *(ontology_entity)_importer_option* hook, which tells the *Ontology Importer* about its functionality. Sub-modules implementing the *(ontology_entity)_importer_option* hook, will be added to the options of a drop-down-list. Upon selecting a sub-module through this element, a sub-form, defined by the sub-module, will be added dynamically to the *Ontology Importer*'s main form. Based on this form, the sub-module also defines appropriate validation and submit hooks, which deal with the actual *Ontology Importer* form validation and submission based on the chosen sub-module. Listing 4.1 shows the basic skeleton of such an *Ontology Importer* sub-module.

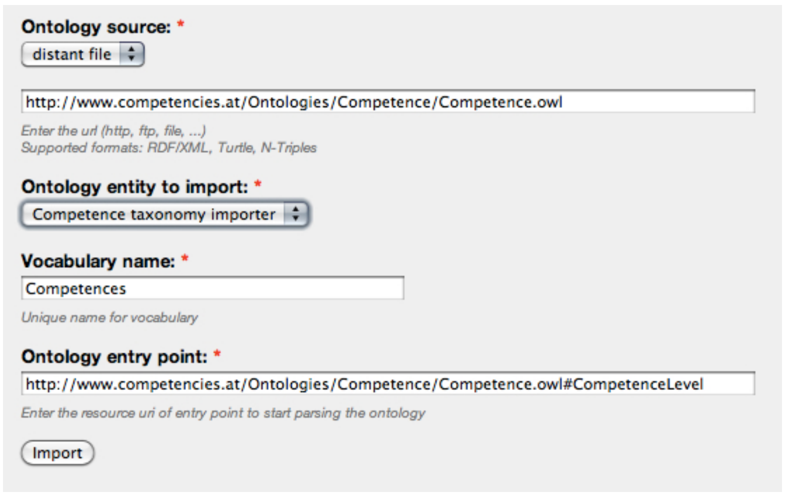
An example implementation of such a sub-module is the *Competence Taxonomy Importer* sub-module for importing a competence term taxonomy into a Drupal taxonomy. It features a sub-form to choose the vocabulary name for the taxonomy to import as well as the ontology entry point. Figure 4.2 depicts the *Ontology Importer* user interface using the *Competence Taxonomy Importer* sub-module.

4.3 Competences

This module implements a CCK field for handling competence data. Instead of dedicating one field per competence, all competence data is stored in one field in serialized form. This field can be used to annotate content nodes with competence data

```
// register ontology entity importer with Ontology Importer module
function example_importer_ontology_entity_importer_option() {
  return array('additional_importer_form' => 'Ontology entity importer');
}

// return Drupal Form API form for importer
function entity_importer_form() {
  ...
  return $form;
}

// validate additional form
function example_importer_form_form_validation($form_state) {
  ...
}

// do something with form
function example_importer_form_submit($form_state) {
  ...
  // parse ontology, do actual import
}
```

Listing 4.1: Basic skeleton of an Ontology Importer sub-module


Ontology source: *

distant file

http://www.competencies.at/Ontologies/Competence/Competence.owl

Enter the url (http, ftp, file, ...)
Supported formats: RDF/XML, Turtle, N-Triples

Ontology entity to import: *

Competence taxonomy importer

Vocabulary name: *

Competences

Unique name for vocabulary

Ontology entry point: *

http://www.competencies.at/Ontologies/Competence/Competence.owl#CompetenceLevel

Enter the resource uri of entry point to start parsing the ontology

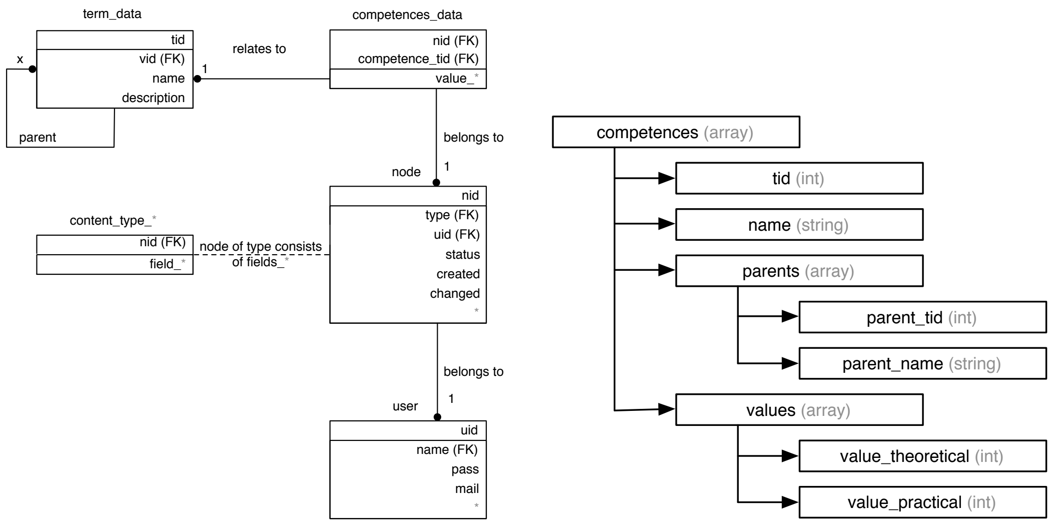
Figure 4.2: Ontology Importer module

building evidences. First data structures used are described followed by implementation details. In the implementation, hooks of interest are chosen and described. The section concludes with a thorough description of the rendering function for the CKK field.

Data Structures

This module, although implementing a CCK field, does not rely on CCK for its table management. Drupal uses an inconsistent data structure for CCK fields: if a field is only present in one content type, it's stored in a *content_type_(type_name)* table as a column. But if it's shared across multiple content types, it moves to its own *content_field_(field_name)* column. Per default this behavior would store all competence data for a specific node in serialized form in only one database field. Instead of using the default method, data is broken down to a per competence level and stored in a dedicated table called *competences_data* (Figure 4.3). This table contains:

- A reference to the node this field belongs to.
- A reference to the term, which describes the competence.
- Columns prefixed with *value_(value_name)*, where *value_name* refers to a human readable name describing the competence characteristics). Per default *value_practical* and *value_theoretical* are used.



(a) Database design (b) Data structure

Figure 4.3: Data structures used in the Competences module

The database schema, especially the part containing the competence values is used to define the possible competence values throughout the system. This is done in order to strengthen system consistency. Other modules could change the value names in this schema through the Drupal Schema API, which provides API functions for creating, dropping, and changing tables, columns, keys, and indexes. Additional competence or competence related values can easily be added using also the Drupal Schema API either directly altering the code in *competences.install* (Listing 4.2) or using the functions *db_add_field*, *db_change_field* and *db_drop_field*. Only requirement is that the column names must be prefixed with *value_*. If applicable, *Competence Broker*s and *Competence Aggregator*s need to to be made aware of additional values. Every following operation on competence data will iterate over the column names in this table containing the prefix *value_*.

```
function competences_schema() {
  $schema['competences_data'] = array(
    ...
    // other database columns
    ...
    'value_practical' => array(
      'type' => 'int',
      'size' => 'tiny',
      'unsigned' => true,
      'default' => -1,
      ),
    // add new value
    'value_new' => array(
      'type' => 'int',
      'size' => 'tiny',
      'unsigned' => true,
      'default' => -1,
    )
  );
  return $schema;
}
```

Listing 4.2: Adding values in *competence.install*

Internally the data obtained from the *competences_data* table for a specific node is used in serialized and unserialized form as a nested associative array. Competence data is joined with additional term data (competence name and, if available, parent terms) from the taxonomy module where the competence term taxonomy is stored (Figure 4.3).

Implementation Details

In *(competences).install* Drupal core hooks are implemented, in a way that the field will be properly added and removed from Drupal and CCK. These hooks tell Drupal to let CCK handle the process through its *content_notify* function. They also handle the creation and removal of the database schema (Figure 4.3a) as well as the definition of global variable defaults for the vocabulary containing the competence terms and the default *Competence Broker* to use. The function *(competences)_field_settings* tells CCK about the settings for this field. This is a sort of catch-all hook for CCK, with several operations. The *form* operation is used to inform CCK about the form for choosing the competence term taxonomy and the *Competence Broker* to use. If the module finds a suitable default variable for these settings, it will default the form elements to it. If the *Competence Broker* features its own settings form, it will be injected into the field's settings form on selection of the *Competence Broker* in a drop-down-list (Figure 4.7a). The function *(competences)_field* tells CCK what to do, in case the field is loaded from or saved to the database or in case the field is deleted. The data structure defined in Figure 4.3b will be composed on loading data from the *competences_data* table and decomposed when inserting or updating to the database. The appropriate functions are invoked in the following three different hooks:

- hook_competences_insert
- hook_competences_update
- hook_competences_delete

By implementing these hooks for database operations, other modules can hook into the system on the storage level when dealing with competence data (e.g. loggers, dashboards, warehousing,...). All three hooks take the same arguments as input:

- An associative array describing the table row to process with as much data columns as possible. (Figure 4.3a)
- A string describing the module invoking the hook.
- A string describing the operation performed (insert, update, delete), to assist possible data analysis later on.

The function *(competences)_generate* implements a hook defined by the contributed *Devel* module, which allows the generation of random data for a field. This implementation of the hook generates random competence data based on the ontology vocabulary to use. This data can be used for testing purposes.

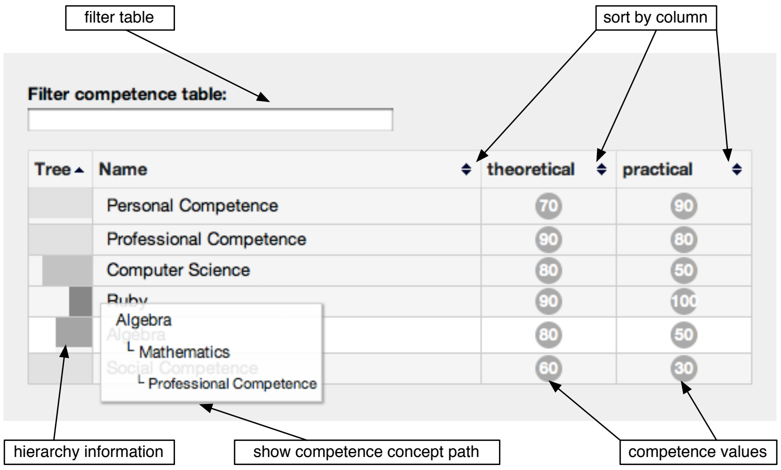
A widget normally defines a default form used to edit a field. This module omits this widget and delegates it to the defined *Competence Broker* module. The function *(competences)_elements* returns an associative array from a callback function, containing data about the *Competence Broker*, the competence term taxonomy, the current value (serialized competence data) and the widget theme function.

In order for the *Competences* field to be properly indexed the *"update index"* operation is intercepted. Every time a Drupal maintenance task updates the index with new node data, a special function is called, searching the published node for updated competence data. It then subsequently adds this competence term to the index. Each competence name will be associated with the node as well as the user authoring the node.

Rendering

A theme function, enveloping a CCK data item is needed to display the widget. Such function defines how the widget looks through different formatters. A default formatter responsible for rendering field data is provided.

In this section the presentation component is a conventional table. Because of the little expressiveness of the competence term taxonomy, a hierarchical approach for profile presentation using a HTML table element as a foundation is used. Each row represents a certain competence. Figure 4.4 depicts the *Competences* field's rendering. Each row represents a certain competence.



Tree ▲	Name	theoretical	practical
	Personal Competence	70	90
	Professional Competence	90	80
	Computer Science	80	50
	Ruby	90	100
	Algebra	80	50
	Social Competence	60	30

Figure 4.4: A sample Competences CKK field

The theme functions incorporate the ideas of the presentation component introduced by Daxböck and Hochmeister (2011). A standard HTML table with additional custom HTML-5 data attributes for each respective column is used and enhanced using Javascript. It consists of the following columns from left to right (Figure 4.4):

Tree By using hierarchical visual cues, the intensity of background color in each first column of a table row is adapted according to how deep a concept resides in the ontology. A tooltip at the left border of each line shows the path in the ontology leading to the concept in reverse order. In order to distinguish two succeeding items on the same level but with different top levels, the two rows are separated with a small gap in this column.

Name The full competence term name.

Values Every competence characteristic is represented by a circled number in this column.

Users also have the ability to personalize their competence profile table by filtering and sorting options. Hence, at the table top, a text-box allowing a user to filter rows which contain a certain string in the full path of a competence. Furthermore, each of the columns in the table provides ascending and descending sorting capability.

The implementation of benefits from using CCK, in that next to the default table representation it also allows other visualizations (e.g. Space-Trees, Hyperbolic-Trees,...) through new widgets and rendering functions to be used. These can be capsuled in their own modules.

4.4 Competence Profile

The *Competence Profile* module implements a CCK field for storing competence profile information. It can be used to implement a content-type representing a competence profile. To very large parts it is similar to the *Competences* CCK field. The reason for providing an own module, is to encapsulate and differentiate evidence competence data from competence profile competence data. Competence profile data also needs additional information about the evidences used to calculate a certain competence in the profile. This module differs in the following points from the *Competences* module:

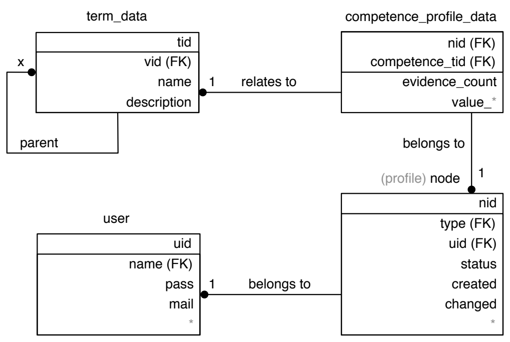
Database

The database schema contains an additional integer field holding the number of evidences, which were used by the *Competence Aggregator* to calculate the competence values in the respective table row. Instead calculating this number on the fly, it is stored in the table to support and possibly speed up certain competence calculation algorithms. Also the internal data structure contains an array storing the ids and names of evidences (Figure 4.5b). Database operations will be done by implementing hooks for inserting, updating and deleting competences in the profile. This again allows other modules to integrate into the field on database level. The appropriate functions are implemented in the three different hooks:

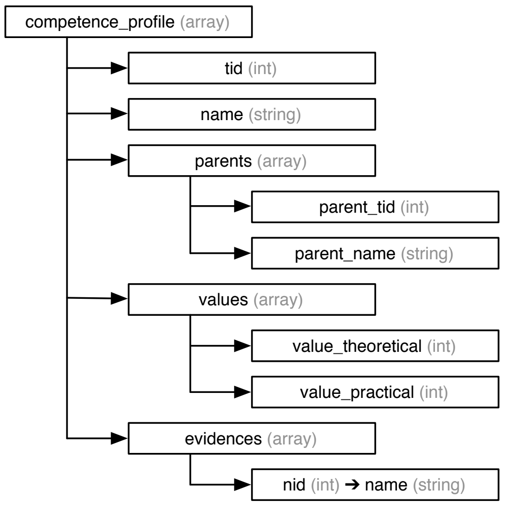
1. hook_competence_profile_competence_insert
2. hook_competence_profile_competence_update
3. hook_competence_profile_competence_delete

Each hook also invokes a special event telling the Rules[4] module that a specific user has just added, updated or deleted a certain competence with certain values. These events can be exploited by the Rules module to trigger certain events (e.g. automatic role assignment, automatic permission assignment,...).

[4]`http://www.drupal.org/project/rules/`, 02.04.2011



(a) Database design



(b) Data structure

Figure 4.5: Data structures used in the Competence Profile module

Settings

The *form* argument of *(competence_profile)_field_settings* tells CCK not only about the competence term taxonomy but also about the *Competence Aggregator* to use. If the module finds a suitable default variable for these settings, it will default the form elements to them. If the *Competence Aggregator* features its own settings form, it will be injected into the field's settings form on selecting the *Competence Aggregator* in a drop-down-list (Figure 4.7b). Also choosable is a set of evidence types. The creation of these types is recommended to a user on an empty competence profile.

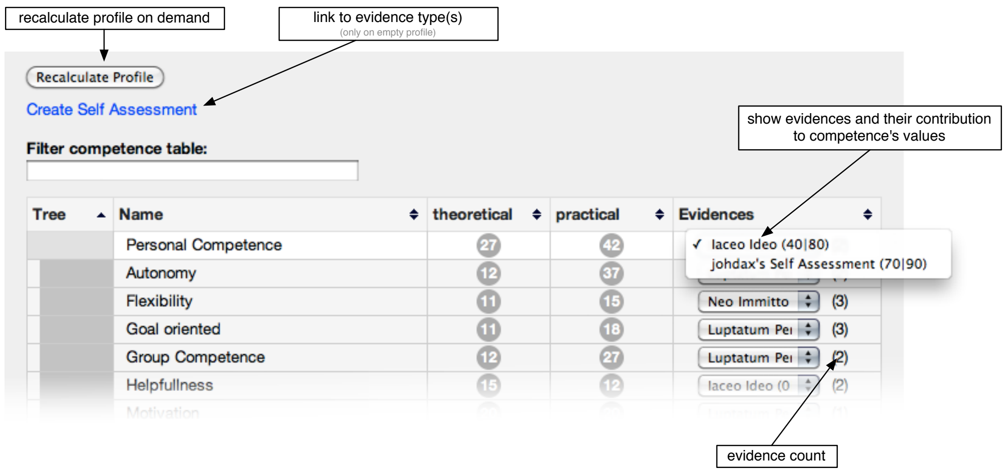
Rendering

Competence profile data is also rendered per default as table, similar to Figure 4.3. It features additional elements like:

- The "Recalculate profile" button on top of the competence table. Clicking on this button invokes the competence calculation algorithm defined in the field's settings, recalculates and dynamically updates the profile using Javascript.
- Additional links to motivate the creation of certain evidence types.
- An additional column to provide evidence information. The column features a dynamical drop-down-list, containing the names of evidences used to calculate this competence's values. Clicking on an entry in the drop down list will redirect to the evidence. Next to the drop down list is the number of evidences used to calculate this competence's values. This

number will also be used to sort the table on this column.

An overview of the competence table rendering can be seen in Figure 4.6.


recalculate profile on demand
link to evidence type(s)
(only on empty profile)
show evidences and their contribution to competence's values
evidence count

Recalculate Profile

Create Self Assessment

Filter competence table:

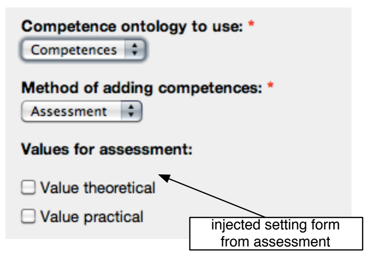
Tree	Name	theoretical	practical	Evidences
	Personal Competence	27	42	✓ Iaceo Ideo (40|80) johdax's Self Assessment (70|90)
	Autonomy	12	37	
	Flexibility	11	15	Neo Immitto (3)
	Goal oriented	11	18	Luptatum Pe (3)
	Group Competence	12	27	Luptatum Pe (2)
	Helpfullness	15	12	Iaceo Ideo (0 (2)
	Motivation			

Figure 4.6: A sample Competence Profile CKK field

Again it should be noted that the implementation as CCK field allows different representation for the same underlying data through new widgets and rendering functions. Possible common representations for competence profiles could be Radar-Diagrams, Star-Models or different kinds of tree representations (Space-trees, Hyperbolic trees,...). These could easily be capsuled in their own respective modules.

Profile export

Export of competence profile data is an important function of the framework. Different export formats are possible, per default a content type containing the *Compe-*



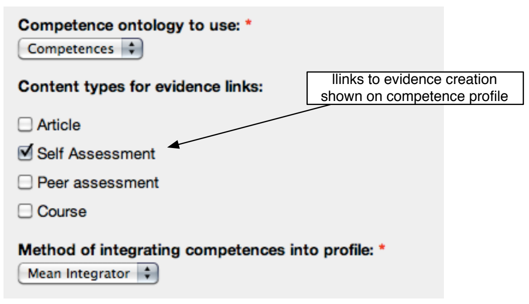
Competence ontology to use: *
Competences

Method of adding competences: *
Assessment

Values for assessment:

☐ Value theoretical
☐ Value practical

(a) Competences



Competence ontology to use: *
Competences

Content types for evidence links:

☐ Article
☑ Self Assessment
☐ Peer assessment
☐ Course

Method of integrating competences into profile: *
Mean Integrator

(b) Competence Profile

Figure 4.7: CCK field settings

tence Profile field can be exported into a serialized PHP-representation of the profile data. However other export formats can easily be added by implementing the appropriate hooks for providing a textual description about the export format and for dealing with the actual export function. The export format is choose-able in the Drupal system settings. Listing 4.3 shows the structure of a simple additional exporter.

```
// implementation of hook_profile_export_format
function example_export_profile_export_format(){
  return array('example_export' => 'example_format');
}

//implemenation of hook_profile_export
function json_export_profile_export($node){
  // competence data data from db
  $competence_data = db_query("SELECT * FROM ({competence_profile_data} p INNER JOIN {term_data} t
       ON p.competence_tid = t.tid) INNER JOIN {node} n ON p.nid = n.nid WHERE p.nid = %d", $node->
       nid);

  // return in export-format
  return export_format_do($competence_data, $node);
}

// actual transformation into export format
function export_format_do($competence_data, $node){
  ...
  // actual transformation algorithm
  ...
}
```

Listing 4.3: Basic skeleton of an example Profile Exporter

4.5 Competence Broker

Competence Broker is the name for the editing widget of the *Competences* module. It is implemented through the *competence_broker* hook and an optional *broker_options* hook. Figure 4.8 shows how a *competence_broker* hook integrates into the edit forms of an evidence node.

A typical *Competence Broker* implementation contains:

- A settings form, which will be injected into the settings form of the *Competences* CKK field.
- An interface for the competence annotation, implemented by the broker taking an evidence's content as input if needed. This interface is added to the field edit form of an evidence (e.g. manual measurement by assessment, automatic competence mining,...).
- A method to add the competence data gathered to the *Competences* CKK field.

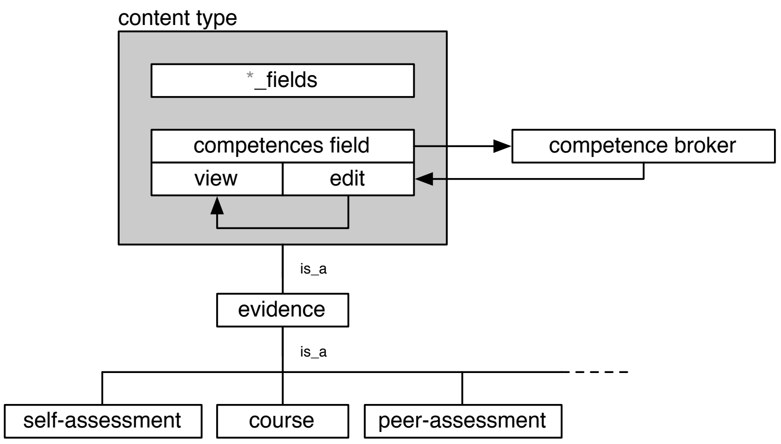


Figure 4.8: Competences field and Competence Broker integration

Listing 4.4 shows the basic skeleton of a *Competence Broker*. In the following an example implementation of a *Competence Broker*, a user interface which can be used for self-, peer- or expert-assessment, is described.

Assessment

This module introduces user interface elements that consist of the following components:

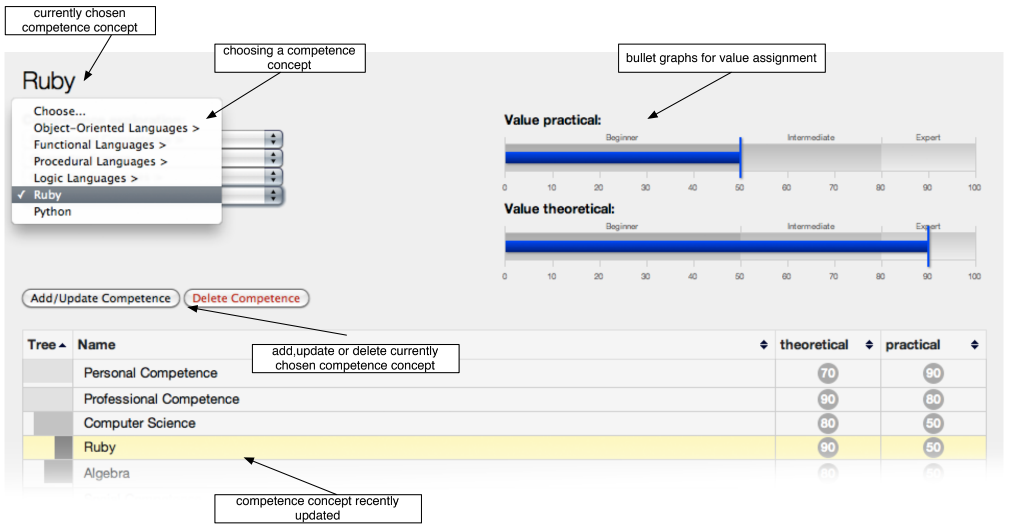
- A navigation component allowing a user to navigate through the competence term taxonomy in order to find a desired competence term.
- An interface element for assigning defined values to a selected competence term.
- Functions to interconnect the competence table with the assessment user interface.

Competence taxonomy navigation

Connected drop-down-lists are used in this module as means of navigation for the competence term taxonomy. Conventional drop-down-lists, showing all available competence terms, are cumbersome for navigation purpose since they do not maintain enough overview over hierarchies. To solve this shortcoming, connected drop-down-lists limit the number of elements to one hierarchy level per drop-down-list. When a user selects a competence term from the list, another drop-down-list will pop up including all child terms from the selected competence above (Figure 4.9). Selecting the option "choose..." causes the lower drop-down-lists to disappear.

```
// callback function for adding competence data
function example_broker_menu() {
  $items['broker_add_competences'] = array(
    'page callback' => 'broker_add_competences_js',
    'access arguments' => array('access content'),
    'type' => MENU_CALLBACK,
  );
  return $items;
}

// button invoking the broker
// implementation of hook_competence_broker
function example_broker_competence_broker($competences_form) {
    $form['group_competences']['example_broker'] = array(
        '#type' => 'submit',
        '#value' => t('Invoke Broker'),
        '#ahah' => array(
          'path' => 'broker_add_competences',
          'wrapper' => 'competence_table_wrapper',
          'method' => 'replace',
          'effect' => 'fade',
          ),
      );
  return $form;
}

// Drupal Form API form for Competence Broker options
// implemention of hook_broker_options
function example_broker_broker_options($value=NULL) {
  ...
  // define form with options
  ...
  return $form;
}

// callback function
// add competences to Competences field
function broker_add_competences_js() {
  // get the form
  $form_state = array('storage' => NULL, 'submitted' => FALSE);
  $form_build_id = $_POST['form_build_id'];
  $form_cache = form_get_cache($form_build_id, $form_state);

  // get ontology
  $ontology  = search_value_by_key($form_cache['#field_info'], 'competence_ontology');

  // use $form_cache['data_field'] to get data from form
  // use user_load() and node_load() function with respective IDs to load additional
      modules for competence mining
  ...
  // get competences
  $competences = get_competences($form_data $ontology);
  ...
  // return updated form
  $output = ahah_render($form, 'competence_table_wrapper');
  return drupal_json(array('data' => $output, 'status' => TRUE));
}
```

Listing 4.4: Basic skeleton of an example Competence Broker


currently chosen competence concept
choosing a competence concept
bullet graphs for value assignment
Ruby
Choose...
Object-Oriented Languages >
Functional Languages >
Procedural Languages >
Logic Languages >
✓ Ruby
Python
Value practical:
Value theoretical:
Beginner
Intermediate
Expert
Add/Update Competence
Delete Competence
add,update or delete currently chosen competence concept
competence concept recently updated

Tree	Name	theoretical	practical
	Personal Competence	70	90
	Professional Competence	90	80
	Computer Science	80	50
	Ruby	90	50
	Algebra	[illegible]	[illegible]

Figure 4.9: Competence term selection and competence value assignment

Competence value assignment

For presentation and modification of competence values, I utilize an adapted graphical element called bullet graph (Few, 2006) as foundation and enhance this element to serve the purpose of competence assessment (Figure 4.10). Basically, the bullet

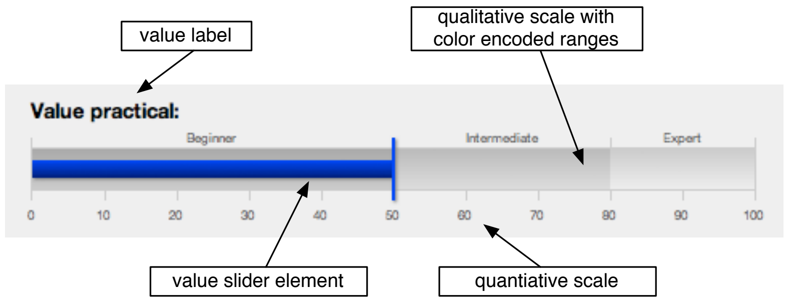


Figure 4.10: Adapted bullet graph for competence assessment

graph consists of a content box including colored ranges representing a qualitative scale, a quantitative scale and a bar reflecting the value. This element is made interactive by enabling a user to drag the value bar to a desired level. A user might also click directly into the content box and the bar will move to this point. On the top of the bullet graph additional labels describe the qualitative scale.

Connected user interface

The assessment navigation component and competences table widget are displayed within the same view as illustrated by Figure 4.9. Therefo the user can search for competences, assess competences and refer to the competence table in one single view. This integrated view adds additional context to the user interface. Moreover, the user interface elements and the competence presentation table are connected as well. Thus, clicking on a table row causes the navigation component to refresh and show the selected competence with its respective values. Recently updated competences in the competence table are highlighted. Adding, updating and deleting are done without page reload utilizing AJAX methods. The competence field is updated locally and stored back to the database on saving the node containing this field.

4.6 Competence Aggregator

Competence Aggregator is the name for the editing widget of the *Competence Profile* module. It is implemented through the *competence_aggregator* hook and an optional *aggregator_options* hook. Figure 4.11 shows how a *Competence Aggregator* hooks into the edit form of a competence profile node.

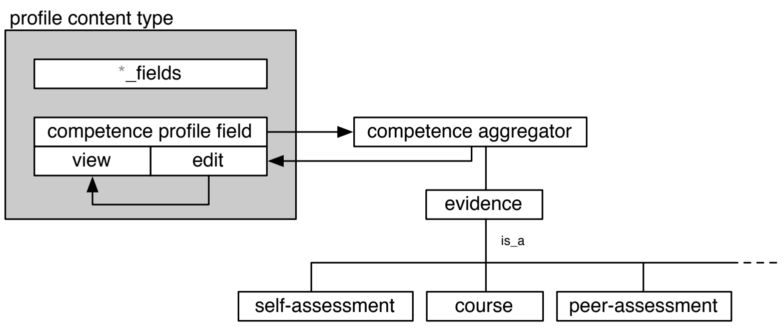


Figure 4.11: Competence Profile field and Competence Aggregator integration

A typical *Competence Aggregator* implementation contains:

- A settings form for an aggregator, which will be injected into the settings form of the *Competence Profile* field.
- An interface providing a method to calculate competences from a user's published evidences taking their competence data as source.
- A method to incorporate the competence profile data to the competence profile CKK field representation.

Listing 4.5 shows the basic skeleton of a *Competence Aggregator*. Following is a sample implementation of a *Competence Aggregator*, which calculates the mean values over all competences stored in a user's evidences.

Mean Aggregator

This module implements the *competence_aggregator* hook. It is used to calculate mean values over all competences, stored in evidences of a certain user. The algorithm workflow is as follows:

1. Invoke the calculation algorithm by clicking the "Recalculate profile" button.
2. The algorithm loops over all competence data stored for published evidences for a certain user. Then it calculates the mean of each respective competence value.
3. The result of this calculation is transformed to an array structure described in Figure 4.5b.
4. The transformed array is then added to the *Competence Profile* field.

It must be noted, that this is just an example implementation, which aim is to show how to build a *Competence Aggregator*. Therefor the actual aggregation process is debatable. This simple calculation is used to test the system's functionality. This thesis won't go into the details of more complex competence aggregation and competence profile calculation. This should be left to further enhancements and additional modules based on the framework.

In addition to the described functionality, COMPETENCE COCKPIT relies on Drupal for taxonomy-, content-, user-, role-, permission- and view-management. This functionality is described in great detail in respective books dealing with Drupal (e.g. (Butcher, 2008; Miles & Miles, 2010; VanDyk, 2008)) and the Drupal website[5]. Therefor descriptions of these functions are omitted in this thesis. However their use in an example system for the evaluation purpose of COMPETENCE COCKPIT is portrayed in the next chapter.

[5] `http://www.drupal.org/`, 05.03.2011

```
// callback function for competence aggregation into a competence profile
function example_aggregator_menu() {
  $items['aggregator_calculate_competences'] = array(
    'page callback' => 'aggregator_calculate_competences_js',
    'access arguments' => array('access content'),
    'type' => MENU_CALLBACK,
  );
  return $items;
}

// define button for competence aggregation
// implementation of hook_competence_aggregator
function example_aggregator_competence_aggregator($competences_form) {
  $form['aggregate_competences']['example_aggregator'] = array(
    '#type' => 'submit',
    '#value' => t('Recalculate Profile'),
    '#ahah' => array(
      'path' => 'aggregator_calculate_competences',
      'wrapper' => 'competence_profile_table_wrapper',
      'method' => 'replace',
      'effect' => 'fade',
    ),
  );
  ...
  return $form;
}

// Drupal Form API form for Competence Aggregator options
// implemention of hook_aggregator_options
function example_aggregator_aggregator_options($value=NULL) {
  ...
  // define form with options
  ...
  return $form;
}

// callback function
// mine competences from node and return updated competences
function aggregator_calculate_competences_js() {
  // get the form
  $form_state = array('storage' => NULL, 'submitted' => FALSE);
  $form_build_id = $_POST['form_build_id'];
  $form_cache = form_get_cache($form_build_id, $form_state);

  // get competence profile owner/user
  $uid = $form_cache['uid']['#value']

  // use user_load() and node_load() function with respective IDs to load additional nodes for
      competence aggregator
  ...
  // get competences
  $competences = calculate_competences($data, $uid);
  ...
  // return updated form
  $output = ahah_render($form, 'competence_table_wrapper');
  return drupal_json(array('data' => $output, 'status' => TRUE));
}
```

Listing 4.5: Basic skeleton of an example Competence Aggregator

Chapter 5

Evaluation

In order to prove that COMPETENCE COCKPIT features the claimed customizability and extendability and that a content management system provides a solid foundation for supporting competence identification, assessment, development and usage, I set up an example competence management system in the university domain. A common approach of developing a specific competence management system would be the following:

1. Design underlying system in Drupal
2. Design competence ontology (terms, characteristics)
3. Define and create content/evidence types
4. Develop and link Competence Brokers for evidence types
5. Define and create content/profile type
6. Develop and link Competence Aggregator for profile type

The system uses the modules described in chapter 4, defines roles and permissions, provides example evidences types with different *Competence Broker*s as well as a content type for competence profiles. Furthermore the user interface takes the defined roles into account and provides access to the management functions for different content types. Additional modules were developed and paths, how to enhance and extend the system further, are shown. The VUT competence ontology provides the base concepts, like the competence term taxonomy to use. The following sections give a detailed overview of the entities of this evaluation system, as well as its possible extensions.

5.1 Preface

The system is built using Drupal 6.20 together with MySQL 5.0.45 and PHP 5.1.6. In addition to the core modules and the modules described in chapter 4 it uses the following contributed modules:

- Menu[1]: allows administrators to customize the site navigation menu. It is used to build the top-level navigation menu.
- Taxonomy[2]: enables the categorization of content. It is used for the administration of the imported competence term taxonomy.
- Content [3] : allows administrators to define new content types. It is used to define evidence types and competence profile types.
- Content permissions [3] : sets content-level permissions for CCK fields. It is used to allow operations on a content type level for different roles.
- Field permissions[4]: sets field-level permissions to allow or omit action on CCK fields for a content type. This granularity allows different fields of the same node to be used by different roles.
- The following are CKK fields used in the evidence and competence profile types.
  - Text [3] : defines CKK text field types.
  - Number [3] : defines CCK numeric field types.
  - Option Widgets [3] : defines CKK selection-, checkbox- and radio-button widgets for text and numeric fields.
  - User Reference [3] : defines a CCK field type for referencing nodes in a field.
  - Date[5]: defines CCK date/time fields and widgets.
- Automatic Node-Titles[6]: allows hiding the content title field and automatic title creation. It is used for evidence and competence profile type title generation to ensure unique and user allocatable title names.
- Publish Content[7]: adds a link to publish and unpublish a node on the node edit/view pages and if enabled in the views listings containing this node. Setting the publishing status of evidences is important, in that only published evidences will be used for competence calculation.
- Content Profile[8]: uses content types for user profiles. Used to link a competence profile type to a user.

[1]http://drupal.org/documentation/modules/menu/, 06.03.2011
[2]http://drupal.org/documentation/modules/taxonomy/, 06.03.2011
[3]http://drupal.org/project/cck/, 06.03.2011
[4]http://drupal.org/project/field_permissions/, 06.03.2011
[5]http://drupal.org/project/date/, 06.03.2011
[6]http://drupal.org/project/auto_nodetitle/, 06.03.2011
[7]http://drupal.org/project/publishcontent/, 06.03.2011
[8]drupal.org/project/content_profile/, 06.03.2011

- Views/Views UI[9]: creates customized lists of content based on database queries defined by the Views UI. Used to create listings of different evidence types.

For this system I rely on the default competence characteristics theoretical knowledge and practical experience level, which in the following will be referred to as theoretical and practical value. No additional competence values are added in this evaluation system.

5.2 Roles and Permissions

Every user in the system can hold several roles. Per default Drupal knows the roles authenticated user and anonymous user. A visitor to the site changes his/her status from an anonymous user to an authenticated one by logging into the system. Per default no content is visible to an anonymous user of the system. Every authenticated user has the permissions to manage its own competence profile, self-assessment and articles. That means, an authenticated user can create, update, delete, publish and unpublish its authored content. In addition to the Drupal default roles the following roles are added.

Peer assessor is allowed to peer assess other users. The user will be the author of a peer-assessment, not the proband. No preconditions for holding certain competence values are given for this role.

Student can manage course content, where the user field of this content references the user of that role. Students are not allowed to add competence data to a course's content.

Lecturer is allowed to manage course content, where the lecturer field references the user of that role. In addition to that lecturers can add competence data to courses where they are referenced in the lecturer field.

Administrator is allowed to manage content types, to import entities of a competence ontology and to administer taxonomies, roles, permissions and users.

The following list gives a description of each permission in the system. Table 5.1 gives an overview of these permissions granted to each role.

- Modules: install and administer additional COMPETENCE COCKPIT modules.
- Users: add, update, delete user profile information.
- Ontology import: import ontology concepts (e.g. competence term taxonomy or value partition).
- Competence term taxonomy: add,update,delete terms in the competence term taxonomy.

[9] http://drupal.org/project/views/, 06.03.2011

- Content types: add,update,delete new content types. This includes adding fields and field settings.
- Menus/Views: add,update,delete new menu items and views/listings of nodes.
- Course(1): (un)publish,add,update,delete courses.
- Course(2): (un)publish and add competence data to courses.
- Articles: (un)publish,add,update,delete articles including the invocation of competence mining.
- Self-assessment: publish and perform or update self-assessment.
- Peer-assessment(1): (un)publish and perform updates on peer-assessment for other users.
- Peer-assessment(2): (un)publish peer-assessment given by other users for the currently logged in user.
- Job profiles: (un)publish,add,update,delete job profiles including adding competence data.
- Competence profile: (un)publish and recalculate a competence profile.
- Profile export: export a competence profile to a defined export format.
- Search: search for content and users.

	anonymous user	authenticated user	admin	peer assessor	student	lecturer
Modules			x			
Users			x			
Ontology import			x			
Competence term taxonomy			x			
Content types			x			
Menus/Views			x			
Course(1)					x	
Course(2)						x
Articles		x				
Self-assessment		x				
Peer-assessment(1)				x		
Peer-assessment(2)		x				

Job profiles		x				
Competence profile		x				
Profile export		x	x	x	x	x
Search		x	x	x	x	x

Table 5.1: Roles and Permissions

Possible enhancements to the roles and permission system could be done by using the Rules module. This module and its integration with the *Competence Profile* module allows building rules, where actions (e.g. adding a role to a user) can be based on certain events (e.g. user holds a competence value greater than a specific value). An example use would be adding an expert role to users if a certain competence value is above a defined threshold, which will subsequently allow this expert to conduct an assessment, which could be valued higher in a specific *Competence Aggregator*, which is aware of expert assessments.

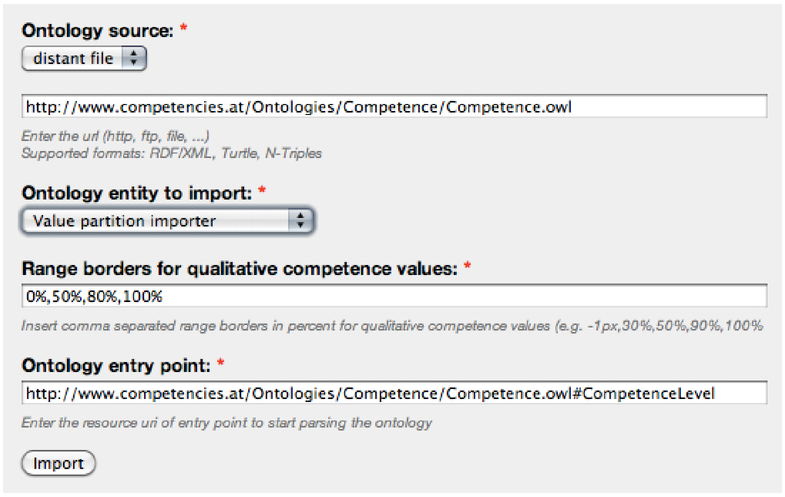
5.3 Ontology Importer

An administrator can import parts of an ontology into a Drupal vocabulary. The competence term taxonomy of the current competence ontology at VUT is imported into a Drupal vocabulary keeping the hierarchical information of the competence terms intact. Figure 4.2 depicts the *Ontology Importer* user interface using the *Competence Taxonomy Importer* sub-module for importing a competence term taxonomy into a Drupal taxonomy. Figure 5.2 shows a small fraction of the resulting imported taxonomy.

Value Partition Importer

Furthermore to show the extendability of the *Ontology Importer* a second sub-module was implemented for importing a competence value partition from the VUT ontology. It features a sub-form to choose the ontology entry point for defining the value partition as well as a text field for defining range borders for this partition. This is a comma separated list of at least 2 values starting with 0 and ending with 100. This is necessary because COMPETENCE COCKPIT maps qualitative to quantitative names, describing a certain range within 0 to 100. The VUT ontology mentioned above doesn't know such a mapping as of writing this thesis. The value partition and its respective ranges are used to update respective system variables. Figure 5.1

depicts the *Ontology Importer* user interface using the *Value Partition Importer* sub-module for importing a value partition to a Drupal variable.



Ontology source: *

distant file

http://www.competencies.at/Ontologies/Competence/Competence.owl

Enter the url (http, ftp, file, ...)
Supported formats: RDF/XML, Turtle, N-Triples

Ontology entity to import: *

Value partition importer

Range borders for qualitative competence values: *

0%,50%,80%,100%

Insert comma separated range borders in percent for qualitative competence values (e.g. -1px,30%,50%,90%,100%

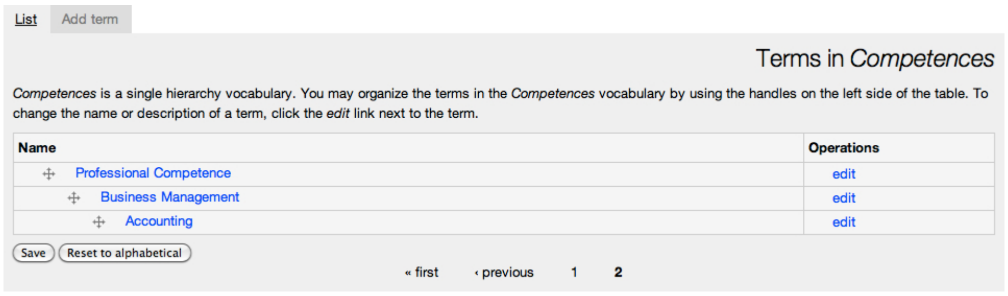
Ontology entry point: *

http://www.competencies.at/Ontologies/Competence/Competence.owl#CompetenceLevel

Enter the resource uri of entry point to start parsing the ontology

Import

Figure 5.1: Value Partition Importer



List | Add term

Terms in *Competences*

Competences is a single hierarchy vocabulary. You may organize the terms in the *Competences* vocabulary by using the handles on the left side of the table. To change the name or description of a term, click the *edit* link next to the term.

Name	Operations
Professional Competence	edit
Business Management	edit
Accounting	edit

Save | Reset to alphabetical

« first ‹ previous 1 2

Figure 5.2: Imported competence term taxonomy

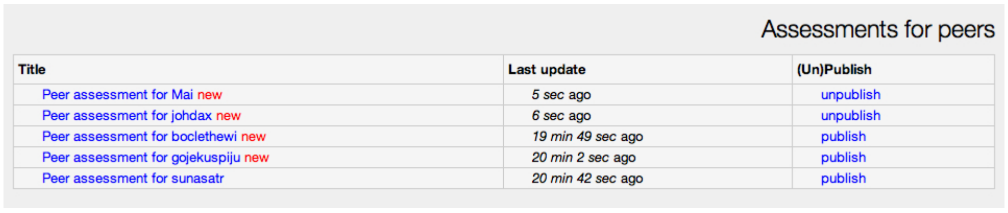
5.4 Evidence Types

Evidences are Drupal content types containing the *Competences* field among other content fields. Only admins are allowed to access the user interface for content type management. Figure 5.5 shows snapshots of this interface responsible for adding and editing content types (Figure 5.5a, Figure 5.5b) together with their respective fields (Figure 5.5c). Users access functions for creating, updating, deleting and publishing content by using two hierarchies of navigation. The top level navigation hierarchy is divided into the sections View, Create and Profile (Figure 5.4):

View provides access to listings of specific content types. If only a single content node of a specific type is available, it links to this content node. Views consist of a tabular listing with a link to a specific content node, the possibility to set a content's publication status as well as, depending on the content type, other columns. Figure 5.3 depicts the view of a user's peer-assessment for a proband.

Create provides access to the pages responsible for the creation of a specific evidence type.

Profile provides access to a user's competence profile. If missing, it creates a competence profile on first use. Also the Logout-Button is located in this section.


Assessments for peers

Title	Last update	(Un)Publish
Peer assessment for Mai new	5 sec ago	unpublish
Peer assessment for johdax new	6 sec ago	unpublish
Peer assessment for boclethewi new	19 min 49 sec ago	publish
Peer assessment for gojekuspiju new	20 min 2 sec ago	publish
Peer assessment for sunasatr	20 min 42 sec ago	publish

Figure 5.3: Peer-assessment view

The second-level navigation is located at top of each content node's form element. Depending on a user's permission it allows to toggle between the View and Edit views of a content node as well as to publish or unpublish a node. Additional access to functions such as saving, deleting or more specifically competence mining or competence calculation are provided directly in the content node's form. Figure 5.4 shows a snapshot of both navigation levels. A more detailed description of each evidence type will be given in the next sections.

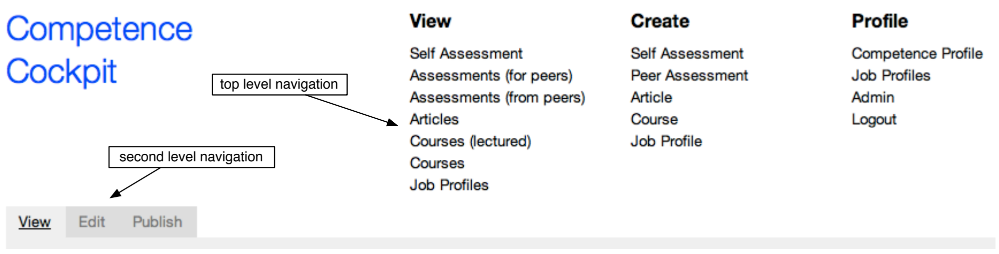
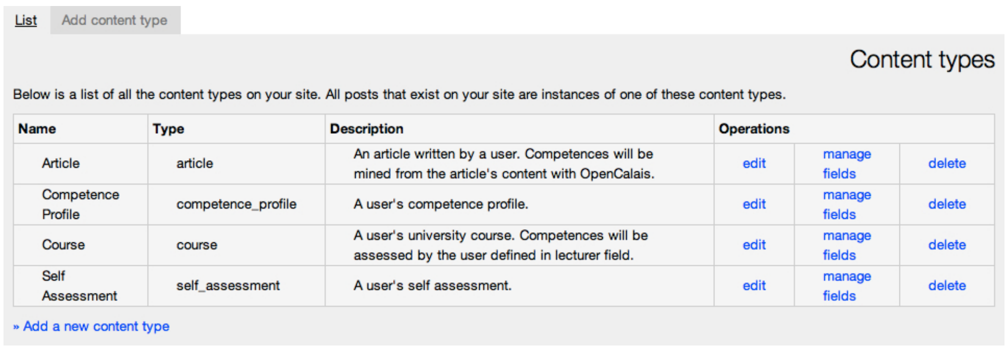


Figure 5.4: Top level and second level navigation



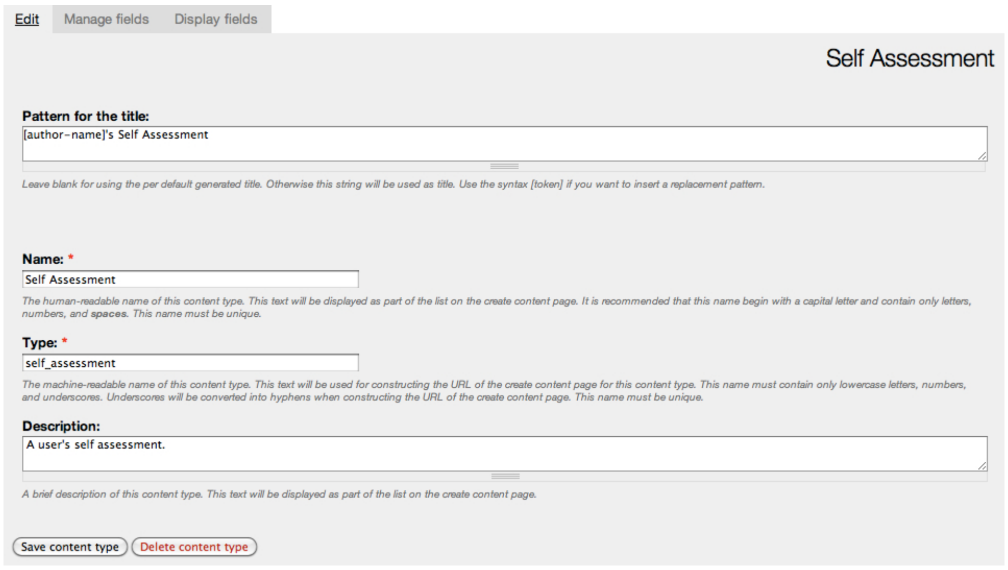
Content types

Below is a list of all the content types on your site. All posts that exist on your site are instances of one of these content types.

Name	Type	Description	Operations		
Article	article	An article written by a user. Competences will be mined from the article's content with OpenCalais.	edit	manage fields	delete
Competence Profile	competence_profile	A user's competence profile.	edit	manage fields	delete
Course	course	A user's university course. Competences will be assessed by the user defined in lecturer field.	edit	manage fields	delete
Self Assessment	self_assessment	A user's self assessment.	edit	manage fields	delete

» Add a new content type

(a) Evidence types



Self Assessment

Pattern for the title:
[author-name]'s Self Assessment

Leave blank for using the per default generated title. Otherwise this string will be used as title. Use the syntax [token] if you want to insert a replacement pattern.

Name: *
Self Assessment

*The human-readable name of this content type. This text will be displayed as part of the list on the create content page. It is recommended that this name begin with a capital letter and contain only letters, numbers, and **spaces**. This name must be unique.*

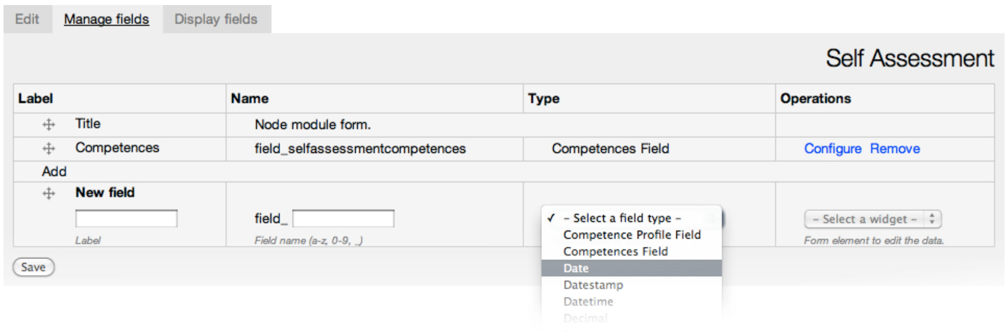
Type: *
self_assessment

The machine-readable name of this content type. This text will be used for constructing the URL of the create content page for this content type. This name must contain only lowercase letters, numbers, and underscores. Underscores will be converted into hyphens when constructing the URL of the create content page. This name must be unique.

Description:
A user's self assessment.

A brief description of this content type. This text will be displayed as part of the list on the create content page.

(b) Content type (Self-assessment) edit



Self Assessment

Label	Name	Type	Operations
Title	Node module form.		
Competences	field_selfassessmentcompetences	Competences Field	Configure Remove
Add			
New field Label	field_ Field name (a-z, 0-9, _)	✓ – Select a field type –	– Select a widget – Form element to edit the data.

(c) Content type (Self-assessment) fields

Figure 5.5: Administer content types user interface

Article

An article is a content type containing a title, a body and *competences* field. It uses the *Simple Miner* module to extract competence data from the content's body field. Competence mining is invoked on demand by clicking on the "Mine competences" button.

Simple Miner

The module introduces a method to mine competences from an evidence's content by implementing the *competence_broker* hook. The mining algorithm itself uses the OpenCalais[10] web service, which ingests unstructured text and uses natural language processing technologies to return results identifying entities within this text. The basic workflow of this module is the following:

1. Invoke the mining algorithm by clicking the "Mine Competences" button.
2. The evidence's textual content is aggregated and preprocessed by stripping unnecessary items and by rewriting problematic characters.
3. The OpenCalais term extraction service is invoked taking the textual representation as input.
4. The OpenCalais term extraction service returns a result of extracted terms, including a relevance score for each term.
5. The XML-formatted result list is transformed to a PHP-array.
6. Each term in the result array is compared with a string similarity measure to each item in the competence term taxonomy. If a term is similar to a taxonomy term, the relevance score will be the value for all competence values for this competence and the competence term name will be added, together with values and parents information to an array described in Figure 4.3b.
7. The resulting array is added to the *Competences* field in serialized form and as table representation.

It must be noted that this implementation is just an example. It however can be used as starting point for research into more advanced mining algorithms! Transforming the term relevance to a competence of the article author is debatable and shows only how a mining algorithm could be implemented using *Competence Broker*. Figure 5.6 shows the process of using the *Simple Miner* module, whereas Figure 5.7 depicts a basic example of a resulting article node after competence mining.

[10] http://www.opencalais.com/,10.04.2011

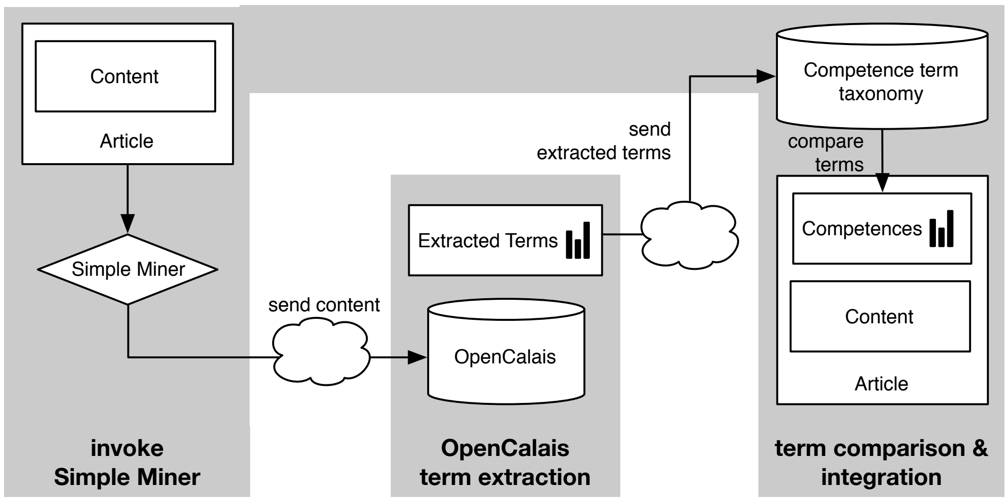
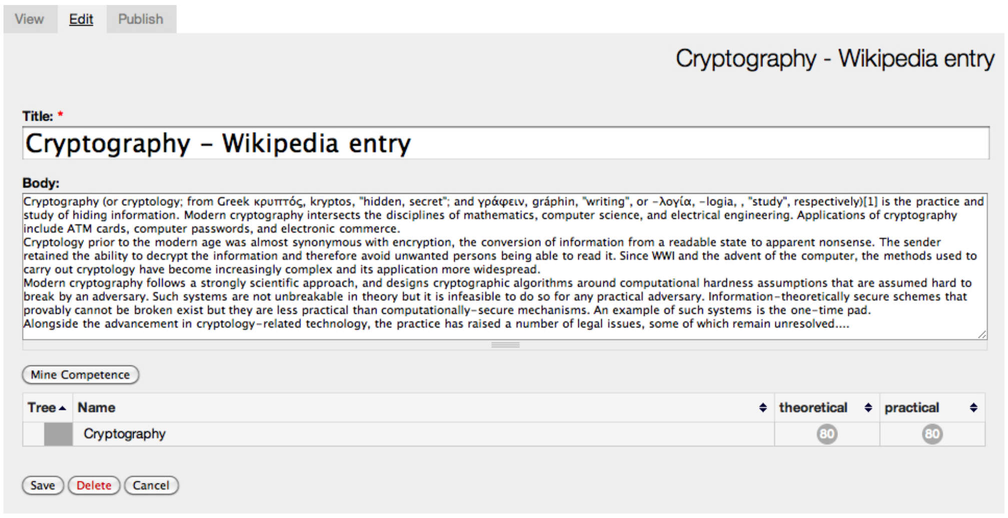


Figure 5.6: Simple Miner process



Cryptography - Wikipedia entry

Title: *

Cryptography – Wikipedia entry

Body:

Cryptography (or cryptology; from Greek κρυπτός, kryptos, "hidden, secret"; and γράφειν, gráphin, "writing", or -λογία, -logia, , "study", respectively)[1] is the practice and study of hiding information. Modern cryptography intersects the disciplines of mathematics, computer science, and electrical engineering. Applications of cryptography include ATM cards, computer passwords, and electronic commerce.
Cryptology prior to the modern age was almost synonymous with encryption, the conversion of information from a readable state to apparent nonsense. The sender retained the ability to decrypt the information and therefore avoid unwanted persons being able to read it. Since WWI and the advent of the computer, the methods used to carry out cryptology have become increasingly complex and its application more widespread.
Modern cryptography follows a strongly scientific approach, and designs cryptographic algorithms around computational hardness assumptions that are assumed hard to break by an adversary. Such systems are not unbreakable in theory but it is infeasible to do so for any practical adversary. Information-theoretically secure schemes that provably cannot be broken exist but they are less practical than computationally-secure mechanisms. An example of such systems is the one-time pad.
Alongside the advancement in cryptology-related technology, the practice has raised a number of legal issues, some of which remain unresolved....

Tree ▲	Name	theoretical	practical
▨	Cryptography	80	80

Figure 5.7: Article example and mined competences

Course

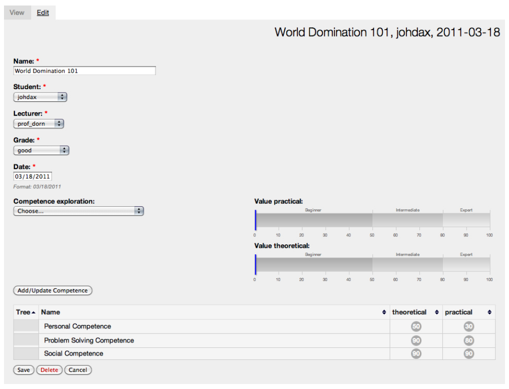
A course is a content type containing information about a university course and the grade a user can obtain from this course. In addition to this grade a course can be annotated with competence data by a lecturer. If this is the case and the course node is published, a course evidence can be used for the competence development process. The content type consists of the field's title, name, student, lecturer, grade, date, competences. This content type comes in different shapes depending on the roles a user holds (lecturer, student).

- The title field is auto-generated with the following format [name],[student],[date] (e.g. "Knowledge Management, Hubert Paul, 10.03.2009").
- The name field contains the course's name in a text field.
- The student field contains a reference to a student, who registered for this course. For the lecturer role this field is choosable. For the student role it is fixed to the student's name. If created by a lecturer, the course will show up in the Courses list of the chosen student, accessible under the View section of the top menu.
- The lecturer field contains a reference to the course lecturer. For the student role this field is choosable. For the lecturer role it is fixed to the student's name. If created by a student, the course will show up in the Courses (lectured) list of the chosen lecturer, accessible under the View section of the top menu.
- On content creation, allowed values for this drop-down-list can be defined. For this example system I chose the traditional Austrian grading system (1-5 / excellent, good, satisfactory, sufficient, unsatisfactory). The value for this field is only choosable by the lecturer role.
- Date is a choosable date of incident, when the grade for this course was obtained. It is choosable by both roles, student and lecturer.
- The competence field contains the competence data associated with this course, which will be used for competence profile calculation of the user referenced by the student field. Only a lecturer is able to add competence data. The *Assessment* module is used to edit the competence data. Both, theoretical and practical values are used.

Figure 5.8 shows two different views of the same course content type for two different roles, the list view for the student role (Figure 5.8a) and the edit view for a lecturer (Figure 5.8b).

Self-Assessment

The self-assessment content type consists of a title and a *Competences* field. The title of a self-assessment will be automatically generated by prefixing the term "Self-Assessment" with the node author's user name. The *Competences* field uses the prac-



World Domination 101, johdax, 2011-03-18

Name: *
World Domination 101

Student: *
johdax

Lecturer: *
prof_dorn

Grade: *
good

Date: *
03/18/2011
Format: 03/18/2011

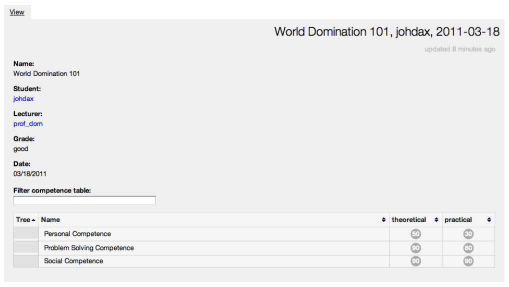
Competence exploration:
Choose...

Value practical:

Value theoretical:

Tree	Name	theoretical	practical
	Personal Competence	50	30
	Problem Solving Competence	90	80
	Social Competence	90	90

(a) Course edit view for lecturer



World Domination 101, johdax, 2011-03-18
updated 8 minutes ago

Name:
World Domination 101

Student:
johdax

Lecturer:
prof_dorn

Grade:
good

Date:
03/18/2011

Filter competence table:

Tree	Name	theoretical	practical
	Personal Competence	50	30
	Problem Solving Competence	90	80
	Social Competence	90	90

(b) Course view for student

Figure 5.8: Course content type for the student role and the lecturer role

tical value and theoretical value as competence values. Method for adding competences is *Assessment*. If no self-assessment is provided by a user, a link for creating a user's self-assessment will be shown under the Create section of the top-menu. After the creation of a user's self-assessment the link for viewing the user's self-assessment moves to the View section of the top-menu. Users can set the publication status of their self-assessment, allowing it contribute to the competence profile calculation. Only one self-assessment per person is allowed. Figure 5.9a depicts an example self-assessment node view.

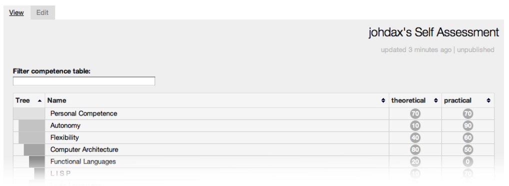
Peer-Assessment

The peer-assessment content type consists of a title, a user reference and a *Competences* field. The user reference field references the proband. The choice of users can be limited to a certain role. Hence the auto-generated title for such a node is the peer's user name prefixed with the term "Peer-Assessment for". The user reference field in this implementation is a select-field but can easily be changed to e.g. an autocompletion-text-field (e.g. for a very large number of users).

Users must possess permissions to create, update and view peer-assessments. That is true for peer-assessments a user has created as well as peer-assessment a user has received.

Under the View section of the top-menu users can find two lists dealing with peer-assessment. One being a list of peer-assessment they have received, the other one being a list of peer-assessments they have created for other users. Each list contains the name of the peer-assessment, which also links to the node, the last update date, the possibility to publish or unpublish a peer-assessments as well as a marker, showing if the content was recently updated.

Users can change the publication status of both types of peer-assessments. Depending on the publication status, the peer-assessment will be used for competence profile calculation or not. Only one peer-assessment for a person is allowed. Figure 5.9b shows an example peer-assessment edit form.



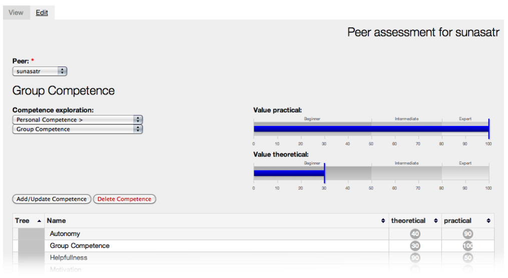
johdax's Self Assessment

updated 3 minutes ago | unpublished

Filter competence table:

Tree	Name	theoretical	practical
	Personal Competence	70	70
	Autonomy	10	90
	Flexibility	40	60
	Computer Architecture	80	50
	Functional Languages	20	0
	LISP	10	70

(a) Self-assessment-node



Peer assessment for sunasatr

Peer: * sunasatr

Group Competence

Competence exploration: Personal Competence > Group Competence

Value practical:

Value theoretical:

Tree	Name	theoretical	practical
	Autonomy	40	90
	Group Competence	30	100
	Helpfullness	90	50
	Motivation	[illegible]	[illegible]

(b) Peer-assessment-node

Figure 5.9: Self/Peer-assessment nodes

Additional Evidence Types

Other evidence types can be added easily using Drupal, CKK or the Views module. These modules allow evidence type creation consisting of different fields, additional navigation items to the menus, listings of types and the type permissions. The Drupal community not only provides a huge number of CCK fields for building different content types, it also provides modules which use external APIs to integrate or expose content from and to Drupal. Since CCK fields can also consist of other nodes or aggregation of nodes (through the View-field module[11]), the possibilities for new content types are not only limitless but can also be added without much or any coding effort.

A possible content type of interest could be a kind of question/answer content type for imitating knowledge market features, like the ones provided by e.g. Quora[12] or StackExchange[13]. Drupal modules containing similar functionality already exist, like the Question module[14] or the Answers module[15]. Every question or answer is stored in its own node. The respective questions and answers can be aggregated with the Views module and further be put into other content types. A special *Competence*

[11]http://drupal.org/project/viewfield/, 26.03.2011
[12]http://www.quora.com/, 05.03.2011
[13]http://www.stackexchange.com/,26.03.2011
[14]http://drupal.org/project/question/, 26.03.2011
[15]http://drupal.org/project/answers/, 26.03.2011

Broker can then be used to assess the competences hidden implicitly in such a conversation and to allocate them to the respective question/answer nodes belonging to a specific users. This methodology would allow the use of a broader scope for competence mining, instead of using a single node.

Another related application could be the use of Activity streams or Actions logs as content type. Action logs capture action like e.g. creating, annotating, accessing, printing, bookmarking, participation,.... Activity streams build a stream for a user by aggregating the social web activities of this user in one place. Whether it's bookmarks, pictures or code commits, posts on a blog, anything created can be gathered and transformed into a single stream. The Activity stream module[16] comes with support for half a dozen popular sites and support for any site that publishes an RSS or Atom feed. Developers can create integrations with any other site using a simple API. Another module implementing a similar feature is the Heartbeat module[17], which allows to display user activity on a website. This module implements an API to log activity. The logged data contains message structures, attributes and variables. Once the activity messages exist in the database, they will be parsed to generate the activity stream. It also integrates into modules for user relationships like described in section 5.5. Competences may be derived from these activities or actions, which may provide supplementary information like the difference between competences in use or in stock. Such an activity stream could be used as content type to integrate an action-based competence capturing approach like described in Lindgren and Stenmark (2002). However such real time capturing needs to be highly user-controllable in order to counteract privacy and monitoring concerns (Lindgren et al., 2004).

Another potential content type could be note content types or idea content types with possible integration with a note-taking, idea-saving application like Evernote[18]. Evernote is used to save ideas from a computer, phone or other device. It syncs notes with a remote server to allow accessing notes on multiple devices. Images will be OCR-ed and made searchable, as will be other file formats. Using an external program like Evernote for information gathering and than expose this information as potential evidence data for COMPETENCE COCKPIT can be a valuable enhancement. The Evernote module[19] can be used as a starting point for such content types.

Another similar content type could be used to aggregate and analyze Twitter[20]- or mail-conversations. Though modules would have to be develop to import or sync with such services.

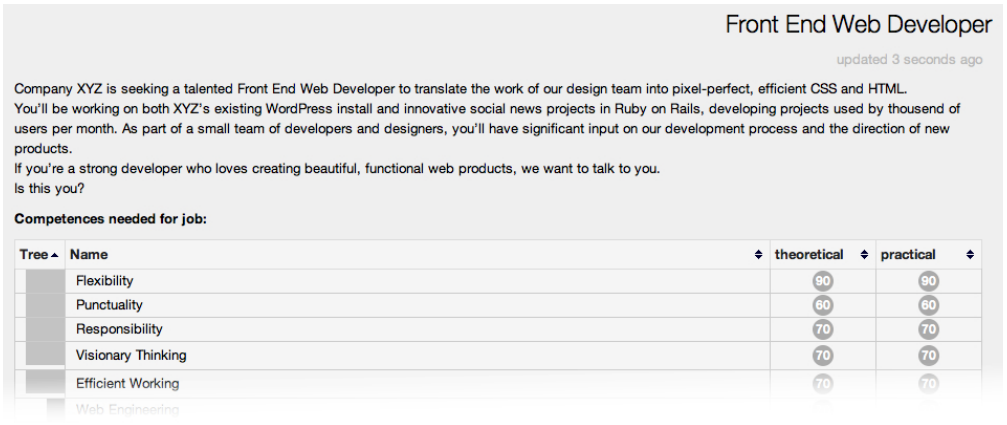
[16]http://drupal.org/project/activitystream/, 26.03.2011
[17]http://drupal.org/project/heartbeat/, 26.03.2011
[18]http://www.evernote.com/, 05.03.2011
[19]http://drupal.org/project/evernote/, 26.03.2011
[20]http://www.twitter.com/, 05.03.2011

5.5 Profile Management

For this example system I implemented two different profile content types. One for representing required profiles (job profile) and one for acquired profiles (competence profile). Acquired profiles specify accomplishments (in terms of competences) of a user, calculated through a user's evidences. Whereas required profiles specify requirements (in terms of competences) to be fulfilled for instance for a certain job.

Job Profile

The job profile content type is used as a required competence profile consisting of a title, a description and a *Competences* field. Title and description are user-choosable and should be used to describe a job. The *Competences* field uses both the practical and theoretical value. As *Competence Broker, Assessment* is used. These annotated competences describe minimum requirements for a job. The nodes of this content type could be used to compare one's competence profile with a job profile. Figure 5.10 depicts an example job profile. A possible enhancement to this content



Front End Web Developer

updated 3 seconds ago

Company XYZ is seeking a talented Front End Web Developer to translate the work of our design team into pixel-perfect, efficient CSS and HTML.
You'll be working on both XYZ's existing WordPress install and innovative social news projects in Ruby on Rails, developing projects used by thousend of users per month. As part of a small team of developers and designers, you'll have significant input on our development process and the direction of new products.
If you're a strong developer who loves creating beautiful, functional web products, we want to talk to you.
Is this you?

Competences needed for job:

Tree ▲	Name ◆	theoretical ◆	practical ◆
	Flexibility	90	90
	Punctuality	60	60
	Responsibility	70	70
	Visionary Thinking	70	70
	Efficient Working	70	70
	Web Engineering		

Figure 5.10: Example job profile

type could be a list of required evidences to further detail a required profile. A similar content type could be used to describe previous and current work experience. Such a content type could add fields for time ranges, links to evidences of projects or work done within such a work experience.

Competence Profile

The competence profile content type consists of a title field, and a *Competence Profile* field. The title is auto-generated by prefixing the term "Competence Profile" with a user name (e.g. Romana's Competence Profile). Additionally the self-assessment content type is chosen to be visible upon an empty profile, to motivate profile population. By using the Content Profile module, this content type will be attached to a user's profile, motivating the creation of a competence profile on user registration. A user can only have one competence profile. The *Competence Profile* field uses the *Mean Aggregator* as *Competence Aggregator*. The competence calculation process is invoked on demand by clicking on the "Recalculate profile" button. Figure 4.6 shows an example competence profile.

Using the *Mean Aggregator* for calculating competence profile data from evidences provides only a simple way of competence calculation. More evidence metadata like (un)learning factor, confidence values, weight or halftime values could be taken into a account when developing a better algorithm for competence profile calculation. Also an additional *Competence Aggregator* could take competence propagation into account or add a kind of recommender system for evidences.

A possible additional value, which could easily be added to a competence profile could be an interest value for competences. Interests or other personal information can alert the context awareness and help in building communities of interests and facilitate indirect communication (e.g. provide information about users with similar interest or competences). For instance Y. Fukami and Takeda (2007) present a method to extract interest value from social bookmarking sites.

In the competence model some notion of interest could also be accommodated at different levels; interest of individuals, organizational interest, group interest. The relationship between interest and competence can be seen as formalized merit, interest as complementary aspect of competence and interest as transcending into competence (Lindgren et al., 2004).

Social Functions

Identity management can be improved by initially motivating the user to join the system without much effort and copious input of profile information. For Drupal there exist a number of methods for authentication with external account data (e.g. TU login [21], OAuth [22], OpenID [23], Facebook Connect [24],...). The Profile module also allows users to not only enter their email address and nickname, but also additional personal data fields (e.g. address-data, phone-numbers, interests, small bio, links to other accounts,...) which can also be integrated from the mentioned other accounts.

Although this work concentrates on the competence of individuals, it is implemented in a way that it's easy to also allow competence profiles for groups or net-

[21] `https://techscreen.tuwien.ac.at/`, 04.04.2011
[22] `http://drupal.org/project/oauth/`, 04.04.2011
[23] `http://drupal.org/project/openid/`, 04.04.2011
[24] `http://drupal.org/project/fbconnect/`, 04.04.2011

works too. The exchangeable *Competence Broker* and *Competence Integration* modules allow easy building and maintenance of competence views different to an individual one (group, core, strategic competence).

To be more useful, users need to be made aware of other's data, through the use of social- and network-functions. More details mean more familiarity and therefor leverage information exchange, competence sharing and building of communities, which again could be used as resources for competence management processes.

Such function could be the definition of relationships and groups. Group administration tasks allow other users to subscribe and unsubscribe to these groups. Selective groups require approval by a group administrator in order to become a member. Also private groups need to be definable. Within informal networks and groups, permissions for member content has to be definable. Group subscribers communicate by using group pages (e.g. aggregations of content of a specific group). After building such groups and aggregated group content, special *Competence Broker*s and *Competence Aggregator*s could be used to make them actionable for competence management. There exist Drupal modules which help implementing the functionality described above and help Drupal to become more social.

- Organic groups [25]: enables users to create and manage their own groups. Each group can have subscribers and maintains a group home page allowing subscribers to communicate with each other. Groups may be selective or not. Selective groups require approval or invitation. Groups can get their own theme, language, taxonomy and integrate well with the Views module. The additional Subgroups[26] module enables a user with the proper permissions to build group hierarchies (or trees) by nesting groups under other groups.
- FriendList[27] and User Relationships[28]: provide both the ability to create custom relationships between users. Admins can create relationship types. Both single (think: a fan) and two-way relationships (think: a friend) are allowed. Both integrate well into other Drupal modules (Views, Rules,...).

A more detailed look at building social networks site with Drupal can be found in Peacock (2009).

Profile export

An additional module for profile export in JSON-format was implemented. Other possible formats could include HR-XML (Allen, 2004) or hResume (King, 2011). This

[25]http://drupal.org/project/og/, 26.03.2011
[26]http://drupal.org/project/og_subgroups/
[27]http://drupal.org/project/friendlist/, 26.03.2011
[28]http://drupal.org/project/user_relationships/, 26.03.2011

could also include the generation of whole resume based on evidence data provided by COMPETENCE COCKPIT. A possible solution for resume generation could be the following procedure:

1. Create evidence types needed (e.g. content types for job descriptions including time ranges, responsibilities, employer information,...)
2. Implement *Competence Broker*s for these evidence types.
3. Mine/Assess competences for these evidence types.
4. Define a resume generation export format (e.g. XML). Iterate over all evidences, competence data and provide evidence and competence profile data in structured data formats.
5. Use XSLT to bring the structured data into different output formats (e.g. PDF, HTML,...).

The provided export could also be used to sync profiles or resume like data with professional web services like LinkedIn[29], for which a Drupal module[30] accessing the LinkedIn API already exists.

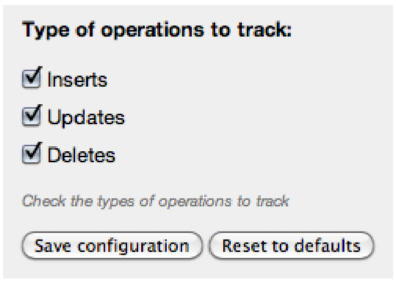
5.6 Competence Warehouse

The *Competence Warehouse* module adds a new table to the database for storing the competence value history of a competence from a competence profile. On every recalculation of the profile, changes will be tracked. This is done by implementing the database operation hooks, defined in the *Competence Profile* field. Every hook responds in a new data row inserted in the competence warehouse table. To make the acquired data more usable, the table contains a couple of additional columns:

- UID: the user-ID of the user to whom the competence profile is assigned.
- NID: the node-ID of the competence profile.
- TID: the term-ID of one competence term.
- Src: the module invoking the hook.
- Type: the exact content type of the competence profile.
- Op: the hook operation called (e.g. insert,update,delete,propagated,...).
- Timestamp: the date and time of incident accurate to the second.
- Value_(value_name): the actual competence values.

The module also contains form settings for selecting which operations to track (Figure 5.11a). Also the module adds a tooltip showing a line-graph of a competence value development history on hovering a column containing the competence value (Figure 5.11b).

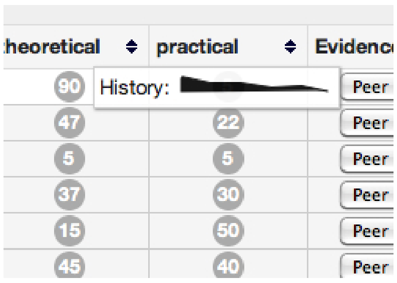
[29] http://www.linkedin.com/, 26.03.2011
[30] http://drupal.org/project/linkedin/, 26.03.2011


Type of operations to track:

- [x] Inserts
- [x] Updates
- [x] Deletes

Check the types of operations to track

(a) Competence Warehouse settings



heoretical	practical	Evidenc
90	History:	Peer
47	22	Peer
5	5	Peer
37	30	Peer
15	50	Peer
45	40	Peer

(b) Competence value history tooltip

Figure 5.11: Competence Warehouse

The data collected by *Competence Warehouse* can further be used for more detailed statistics. For instance the Quant module[31] is a Drupal module based engine for producing quantitative, time-based analytics for any Drupal component. Quant takes raw data from regular Drupal actions (e.g. node creation, node update,...) and plots the activity over a specific time-range. This functionality can be used by other modules, like *Competence Warehouse* for providing charts about evidence creation or specific competence development. Data can also be aggregated, for instance over groups of users, to analyze the competence development over groups. External modules can use Quant by implementing *hook_quant*, which returns, along other information, an array containing the database table to watch and the field containing timestamp information.

5.7 Search

Drupal adds a module responsible for search and retrieval functionality by default. Users can search for users and particular terms in content nodes. On the content-tab of the search page, users are able to search for terms appearing in the default rendering of a node's content on a site, which also includes the rendering of any CCK fields (hence also *Competences* and *Competence Profile* fields). On the users-tab of the search page, users are able to search user names of registered users on a site.

In addition to that, every competence, which shows up in the competence profile of a user, is also associated with the user search type and a competence profile search type. This can allow to search and find users based on a certain competences. However a user interface exploiting this index is not yet available and scheduled as future

[31] http://drupal.org/project/quant/, 26.03.2010

extension. Instead the standard full text engine, splitting the competence data into terms is used to search for content containing certain competences.

During content search, if users enter more than one search term the search module will look for content that has all the terms entered. If instead either one term or another term is wanted, terms can be joined with "or". Looking for an exact phrase can be done by enclosing it in quotation marks.

Drupal's search engine indexes the text content of a site. The search engine does its indexing at intervals using cron.

Content-related actions on a site (creating, editing, or deleting) automatically cause affected content items to be marked for indexing or reindexing at the next cron run. When content is marked for reindexing, the previous content remains in the index until cron runs again.

There also exist additional modules for further refining search, from changing the underlying search engine (e.g. ApacheSOLR[32]) to adding faceted search[33] functionality. These modules can be used as basis if a much more detailed and refined search experience for COMPETENCE COCKPIT which also takes the special user type and competence profile type indices into account.

5.8 Semantic Integration

Providing deeper semantic integration allows linking COMPETENCE COCKPIT to the Web of Data. Breslin et al. (2007) already present a conceptual framework using Friend of a Friend (FOAF), Semantically Interlinked Online Communities (SIOC) and Simple Knowledge Organization System (SKOS) vocabularies for finding experts with desired expertise in a domain of interest. All three vocabularies could be integrated into COMPETENCE COCKPIT. The FOAF ontology has properties, which define topics of interest to a person. There also exist extensions to the FOAF ontology for describing career information or resume type information like the FOAF resume schema (Bojars, 2004) or the Description of a Career (DOAC)[34] vocabulary. There also already exist a FOAF module[35] for Drupal which allows to:

- automatically import/synchronize profile information between any Drupal-powered sites,
- import profile information from external FOAF files and
- export a FOAF file based on a user's profile.

[32]drupal.org/project/apachesolr/, 26.03.2010
[33]drupal.org/project/faceted_search/, 26.03.2010
[34]http://ramonantonio.net/doac/ 3.3.2011
[35]http://drupal.org/project/foaf/ 3.3.2011

The SIOC project is an open specification for describing communities using online discussion communities. Allowing the integration of SIOC properties could support the interlinkage between COMPETENCE COCKPIT evidence types and other community content. For SIOC too a Drupal module[36] exists. Finally the SKOS ontology could be linked with the Drupal taxonomy module, allowing the alignment of competence ontology concepts with SKOS or align content types with SKOS.

Such semantic integration needs to avoid unnecessary duplication when using different vocabularies for COMPETENCE COCKPIT. But a clear integration could further strengthen the reusability of the data stored in COMPETENCE COCKPIT as well as facilitate the easier integration of outside content.

[36] http://drupal.org/project/sioc/ 3.3.2011

CHAPTER 6

CONCLUSION AND FUTURE WORK

In this thesis I argue that a modern content management system, like Drupal, can be extended to support the competence management processes: (1) competence identification, (2) competence measurement, (3) competence development and (4) competence usage. They are suited as foundation for a modular and flexible competence management framework. Also its use lowers the entry barrier for competence management and makes it more unobtrusive and maintainable.

I presented a competence model, handling different resources, which can be made actionable for competence management. This making actionable is achieved by providing interfaces for different methods for competence measurement, tailored to the needs of the specific resources. Furthermore competence profiles can be calculated and generated by using these resources and their annotated competence data as evidences.

This model is then transformed into modules for the Drupal content management system, bundling the described functionality of the model's entities. Modules for competence data and competence profile data representation, competence assessment, competence mining and competence profile calculation as well as modules for importing concepts from an existing competence ontology were developed. The content management functions of Drupal were used to create different evidence types, showing that it's easy to plug into already exiting data for competence measurement, development and usage.

The implemented modules based on Drupal were put to test in setting up an example system in the university domain. The extendability was evaluated by implementing additional modules and further showing paths to extend the framework.

COMPETENCE COCKPIT provides a solid foundation for implementing specific competence management system. It allows to focus such system development on detailing core competence management functions like meaningful competence mining or calculation algorithms. Also its modularity facilitates the adoption of new research methods in the above named processes of competence management. The addition of new evidence types, *Competence broker*s and *Competence Aggregator*s can be accomplished without much effort. Also taking social functions, provided by the underlying content management system and other levels of competence management (groups, networks, organizations,...) into account can further improve such a system.

A main shortcoming of COMPETENCE COCKPIT is a missing similarity measure framework, which can be used to compare competence data on different levels. Such a framework can feature some kind of gap analysis for evidences and profiles. This may either be the comparison between acquired and required profiles (e.g. compare the competence profile of a user with a job description profile) or two required profiles (e.g. compare two user profiles). In addition to that, it could be used to also compare evidences on a competence data level, groups of people or different *Competence broker*s and *Competence Aggregator*s. One assumption is, that he competence usage process can benefit greatly from allowing aggregation and comparison on different levels.

Also search and retrieval functions can be improved and enhanced making it easier to find individuals or groups which hold certain competences. The implementation of competence data and competence profile data benefits from using CCK, in that it allows different visualizations (e.g. Radar-Diagrams, Space-Trees, Hyperbolic-Trees,...) to be used, which can be capsuled in their own modules.

Another main point of improvement, which would need existing module code to be rewritten, is stronger semantic integration. There already exist different vocabularies, which can be integrated into Drupal based systems and which can be exploited for competence management. Being able to connect with such vocabularies would give COMPETENCE COCKPIT a welcomed integration in the Web of Data. A framework for semantic integration, allowing easy existing vocabulary integration, based on the already existing *Ontology Importer* module, seems like a worthwhile addition to the framework.

Last but not least the framework should be evaluated within different domains, preferably in real corporate and organizational environments, which may already use Drupal in their daily routines. Already existing module extensions like the *Competence Warehouse* module, which was already used for evaluation purposes by Daxböck and Hochmeister (2011) can provide the necessary foundation for this task.

The claim, that this framework supports the development and integration of specific competence management systems into daily work activities must be based on solid scientific research.

References

Allen, C. (2004). *Competencies 1.0 (Measurable Characteristics)*. Retrieved 01.03.2011, from xml.coverpages.org/HR-XML-Competencies-1_0.pdf

Baladi, P. (1999). Knowledge and competence management: Ericsson business consulting. *Business Strategy Review*, *10*, 20–28.

Becerra-Fernandez, I. (2006). Searching for experts on the Web: A review of contemporary expertise locator systems. *ACM Transactions on Internet Technology*, *6*(4), 333–355.

Berio, G., & Harzallah, M. (2005). Knowledge management for competence management. *Journal of Universal Knowledge Management*, 21–28.

Berlanga, A., Bitter-Rijpkema, M., Brouns, F., Sloep, P., & Fetter, S. (2009). Personal Profiles: Enhancing Social Interaction in Learning Networks. *International Journal of Web Based Communities*, *7*(1), 66–82.

Biesalski, E., & Abecker, A. (2005). Human Resource Management with Ontologies. *Professional Knowledge Management*, 499–507.

Bojars, U. (2004). Extending FOAF with Resume Information. In *1st Workshop on FOAF, Social Networks and the Semantic Web*. Galway, Ireland.

Boyatzis, R. (1982). *The competent manager: A model for effective performance*. John Wiley & Sons Inc.

Braun, S., Kunzmann, C., & Schmidt, A. (2010). People Tagging and Ontology Maturing: Toward Collaborative Competence Management. *From CSCW to Web 2.0: European Developments in Collaborative Design*, 133–154.

Breslin, J., Bojars, U., Aleman-Meza, B., Boley, H., Mochol, M., Nixon, L., et al. (2007). Finding experts using Internet-based discussions in online communities and associated social networks. In *1st International ExpertFinder Workshop*. Berlin, Germany.

Butcher, M. (2008). *Learning Drupal 6 Module Development: A practical tutorial for creating your first Drupal 6 modules with PHP*. Packt Publishing.

Cohen, D., & Prusak, L. (2001). *In good company: How social capital makes organizations work*. Harvard Business Press.

Colucci, S., Di Noia, T., Di Sciascio, E., Donini, F., & Ragone, A. (2007). Measuring core competencies in a clustered network of knowledge. In *Proceedings of the International Conference on Knowledge Management* (pp. 279–291). Vienna, Austria.

Corlosquet, S., Delbru, R., Clark, T., Polleres, A., & Decker, S. (2009). Produce and Consume Linked Data with Drupal! In *Proceedings of the 8th International Semantic Web Conference* (pp. 763–778). Washington DC, USA.

Crowder, R., Wilson, M., Fowler, D., Shadbolt, N., Wills, G., & Wong, S. (2009). Navigation Over a Large Ontology for Industrial Web Applications. San Diego, USA.

Daxböck, J., & Hochmeister, M. (2011). A User Interface for Semantic Competence

Profiles. In *19th International Conference on User Modeling, Adaptation, and Personalization.* Girona, Spain.

De Coi, J., Herder, E., Koesling, A., Lofi, C., Olmedilla, D., Papapetrou, O., et al. (2007). A model for competence gap analysis. In *Proceedings of the 7th International Conference on Web Information Systems and Technologies.* Noordwijkerhout, The Netherlands.

Dolog, P., & Schäfer, M. (2005). A framework for browsing, manipulating and maintaining interoperable learner profiles. In *Proceedings of the 10th International Conference on User Modelling* (pp. 397–401).

Dorn, J. (2010). *Competence Measurement.* Retrieved 3.3.2011, from http://www.competencies.at/node/170

Dorn, J., Herzog, C., & Werthner, H. (2008). TechScreen: Networked Knowledge Management. In *Smart Business Networks* (pp. 460–470). Bejing, China.

Dorn, J., & Hochmeister, M. (2009). TechScreen: Mining Competencies in Social Software. In *Proceedings of the 13th World Multi-Conference on Systemics, Cybernetics and Informatics* (pp. 115–126). Orlando, USA.

Dorn, J., Naz, T., & Pichlmair, M. (2007). Ontology development for human resource management. In *Proceedings of 4th International Conference on Knowledge Management* (pp. 109–120). Vienna, Austria.

Dorn, J., & Pichlmair, M. (2007). A Competence Management system for Universities. In *Proceedings of European Conference on Information Systems* (pp. 759–770). St. Gallen, Switzerland.

Dorn, J., Pichlmair, M., Schimper, K., & Tellioğlu, H. (2008). Supporting Competence Management in Software Projects. In *Proceedings of International Conference on Concurrent Enterprising* (pp. 451–458). Lisboa, Portugal.

Draganidis, F., Chamopoulou, P., & Mentzas, G. (2008). A semantic web architecture for integrating competence management and learning paths. *Journal of Knowledge Management*, *12*(6), 121–136.

Draganidis, F., & Mentzas, G. (2006). Competency based management: a review of systems and approaches. *Information Management & Computer Security*, *14*(1), 51–64.

Dubois, D., Shadden, M., Kaufman, R., & Brethower, D. (2000). The competency casebook: Twelve studies in competency-based performance improvement. *Performance Improvement*, *39*(1), 37–40.

Ehrlich, K., Lin, C., & Griffiths-Fisher, V. (2007). Searching for experts in the enterprise: combining text and social network analysis. In *Proceedings of the International ACM Conference on Supporting Group Work* (pp. 117–126). Sanibel Island, USA.

Erpenbeck, J., & Rosenstiel, L. von. (2003). *Handbuch Kompetenzmessung: Erkennen, verstehen und bewerten von Kompetenzen in der betrieblichen, pädagogischen* *und psychologischen Praxis.* Schäffer-Poeschel.

Farrell, S., Lau, T., Nusser, S., Wilcox, E., & Muller, M. (2007). Socially augmenting employee profiles with people-tagging. In *Proceedings of the 20th annual ACM Symposium on User Interface Software and Technology* (pp. 91–100). Newport, USA.

Few, S. (2006). *Information dashboard design: the effective visual communication of data.* O'Reilly Media, Inc.

Harzallah, M., Berio, G., & Vernadat, F. (2005). Analysis and modeling of individual competencies: toward better management of human resources. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, *36*(1), 187–207.

Hong, J., & Ståhle, P. (2005). The coevolution of knowledge and competence management. *International Journal of Management Concepts and Philosophy*, *1*(2), 129–145.

Hustad, E., & Munkvold, B. (2005). IT-supported competence management: a case study at Ericsson. *Information Systems Management*, *22*(2), 78–88.

Jackson, S., & Schuler, R. (2003). *Managing human resources through strategic partnerships.* Thomson/South-Western.

Jiang, T., & He, D. (2007). Redefining Social Network Services: A Solution to Personal Information and Knowledge Management. In *Proceedings of the 2007 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology - Workshops* (pp. 292–295). Washington DC, USA.

Katifori, A., Halatsis, C., Lepouras, G., Vassilakis, C., & Giannopoulou, E. (2007). Ontology visualization methods - a survey. *ACM Computing Surveys*, *39*(4).

King, R. (2011). *Microformats Draft Specification.* Retrieved 26.03.2011, from `http://microformats.org/wiki/hresume`

Knight, C., Gasevic, D., & Richards, G. (2005). Ontologies to integrate learning design and learning content. *Journal of Interactive Media in Education*, *7*(7).

Kokko, N., Vartiainen, M., & Lönnblad, J. (2007). Individual and collective competences in virtual project organizations. *The Electronic Journal for Virtual Organizations and Networks*, *8*, 28–52.

Lahti, R. (1999). Identifying and integrating individual level and organizational level core competencies. *Journal of Business and Psychology*, *14*(1), 59–75.

Liao, M., Hinkelmann, K., Abecker, A., & Sintek, M. (1999). A competence knowledge base system as part of the organizational memory. In *Proceedings of the 5th biannual german conference on knowledge based systems* (pp. 125–137). W "urzburg, Germany.

Lindgren, R., Henfridsson, O., & Schultze, U. (2004). Design principles for competence management systems: a synthesis of an action research study. *MIS quarterly*, *28*(3), 435–472.

Lindgren, R., & Stenmark, D. (2002). Designing Competence Systems. *Scandinavian Journal of Information Systems*, *14*(1), 19–35.

Lindgren, R., Stenmark, D., & Ljungberg, J. (2003). Rethinking competence systems for knowledge-based organizations. *European Journal of Information Systems*, *12*(1), 18–29.

Marenzi, I., Demidova, E., Nejdl, W., & Zerr, S. (2008). Social software for lifelong competence development: Challenges and infrastructure. *International Journal of Emerging Technologies in Learning*, *3*, 13–18.

McClelland, D. (1973). Testing for competence rather than for intelligence. *American psychologist*, *28*(1), 1–14.

Miles, E., & Miles, L. (2010). *Drupal's Building Blocks: Quickly Building Web Sites with CCK, Views, and Panels*. Addison-Wesley Professional.

Nonaka, I., & Takeuchi, H. (1995). *The Knowledge-Creating Company. How Japanese Companies Create the Dynamics of Innovation*. Oxford University Press.

Oliveira, J., Souza, J. de, Miranda, R., Rodrigues, S., Kawamura, V., Martino, R., et al. (2006). GCC: a knowledge management environment for research centers and universities. In *Proceedings of Frontiers of WWW Research and Development - APWeb* (pp. 652–667). Harbin, China.

Paquette, G. (2007). An ontology and a software framework for competency modeling and management. *Educational Technology Educational Technology Society*, *10*(3), 1–21.

Peacock, M. (2009). *Drupal 6 Social Networking*. Packt Publishing.

Prahalad, C., & Hamel, G. (2006). The core competence of the corporation. *Strategische Unternehmungsplanung - Strategische Unternehmungsführung*, 275–292.

Rodrigues, S., Oliveira, J., & Souza, J. de. (2005). Competence mining for team formation and virtual community recommendation. In *Proceedings of the 9th International Conference on Computer Supported Cooperative Work in Design* (pp. 44–49). Coventry University, UK.

Sanchez, R. (2003). *Knowledge management and organizational competence*. Oxford University Press, USA.

Scarbrough, H. (1998). Path(ological) dependency? Core competencies from an organizational perspective. *British Journal of Management*, *9*(3), 219–232.

Schmidt, A., & Kunzmann, C. (2006). Towards a human resource development ontology for combining competence management and technology-enhanced workplace learning. In *Workshops on the move to meaningful Internet Systems* (pp. 1078–1087). Montpellier, France.

Shneiderman, B. (1996). The eyes have it: A task by data type taxonomy for information visualizations. In *Proceedings of the IEEE Symposium on Visual Languages* (pp. 336–343). Washington DC, USA.

Sieg, A., Mobasher, B., & Burke, R. (2007). Web search personalization with ontological user profiles. In *Proceedings of the 16th ACM Conference on Information and Knowledge Management* (pp. 525–534). Napa Valley, USA.

Smith, B. (2003). Ontology: an introduction. *Blackwell Guide to the Philosophy of Computing and Information*, 155–166.

Stenmark, D., & Lindgren, R. (2004). Integrating knowledge management systems with everyday work: design principles leveraging user practice. In *Proceedings of the 37th Annual Hawaii International Conference on System Sciences* (pp. 5–7). Big Island, USA.

Storey, M., Musen, M., Silva, J., Best, C., Ernst, N., Fergerson, R., et al. (2001). Jambalaya: Interactive visualization to enhance ontology authoring and knowledge acquisition in Protégé. In *Workshop on Interactive Tools for Knowledge Capture.* Victoria, Canada.

Trichet, F., Bourse, M., Leclere, M., & Morin, E. (2004). Human resource management and semantic Web technologies. In *Proceedings of the International Conference on Information and Communication Technologies: From Theory to Applications* (pp. 641–642). Damascus, Syria.

VanDyk, J. (2008). *Pro Drupal Development.* Apress.

Velardi, P., Cucchiarelli, A., & Petit, M. (2007). A Semantically Enriched Competency Management System to Support the Analysis of a Web-based Research Network. In *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence* (pp. 41–47). Silicon Valley, USA.

Willett, W., Heer, J., & Agrawala, M. (2007). Scented widgets: Improving navigation cues with embedded visualizations. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 51–58). San Jose, USA.

Woelk, D. (2002). E-learning, semantic web services and competency ontologies. In *Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications* (pp. 2077–2078). Denver, USA.

Y. Fukami, I. O., T. Sekiya, & Takeda, H. (2007). Method of Evaluating Contents on the Basis of Community's Interest Using Data from Social Bookmark Services. In *Proccedings of the 6th International Workshop on Social Intelligence Design* (pp. 91–205). Trento, Italy.

Zemke, R. (1982). Job Competencies: Can They Help you Design Better Training? *Training, 19*(5), 28–31.