LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN
Institut für Informatik
Lehr- und Forschungseinheit für
Programmierung und Softwaretechnik

Diploma Thesis

Ontology-Based Development and Promotion of Acceptance of an Intranet Infrastructure

Diplomarbeit

Ontologiebasierte Entwicklung und Akzeptanzförderung einer
Intranet Infrastruktur

Thomas L. Lederer
tom@lederer.it

LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN
Institut für Informatik
Lehr- und Forschungseinheit für
Programmierung und Softwaretechnik

Diploma Thesis

Ontology-Based Development and Promotion of Acceptance of an Intranet Infrastructure

Diplomarbeit

Ontologiebasierte Entwicklung und Akzeptanzförderung einer Intranet Infrastruktur

Thomas L. Lederer
tom@lederer.it

Begin of processing period:	December, 1st 2006
End of processing period:	May, 31st 2007

Responsible professor and internal supervisor:

Prof. Dr. Rolf Hennicker	LMU München	Rolf.Hennicker@ifi.lmu.de

External supervisors:

Bernd Domansky	Sun Microsystems	Bernd.Domansky@Sun.COM
Christine Jeske	Sun Microsystems	Christine.Jeske@Sun.COM

Abstract

Ontology-Based Development and Promotion of Acceptance of an Intranet Infrastructure

This thesis consists of the generation of two ontologies. The first one was motivated by the need to consolidate intranet pages and to select a new infrastructural platform that would also allow to be adapted to future needs. As a tool to help making the decision amongst the many different software solutions that exist, an ontology was be used. The according ontology was built from the parameters that exist in the company and that would be influential to the decision. The properties of the elements of an ontology should allow to relate elements like web server, skills, employees to each other, so that in the end a logical deduction can be drawn, which solutions fits the prerequisites best. The relationships are depicted in images showing the connections between ontology elements. But while individual connections can be examined well, larger networks turned out to be difficult to read and interpret.

The second ontology deals with Sun Microsystems' "Change Acceptance Program", which is a program to increase acceptance of newly introduced software, tools or workflows. Even though the official manual brings several (paper) tools, the idea to use an ontology as status monitor emerged. The current status of important factors can be recorded by it (e.g. stance of persons toward the change). As a result of this thesis, it became clear that ontologies can be used to keep track of a CAP, but the effort to use and maybe learn new software to edit and use the ontologies most likely cancels the benefits that comes with using ontologies.

Zusammenfassung

Ontologiebasierte Entwicklung und Akzeptanzförderung einer Intranet Infrastruktur

Im Laufe dieser Diplomarbeit wird die Erzeugung von zwei Ontologien behandelt. Die erste wurde motiviert durch die Notwendigkeit einer Konsolidierung von Intranetseiten und die Auswahl einer neuen Infrastrukturplattform, die sich auch zukünftigen Anforderungen anpassen lassen würde. Die Ontologie wurde als ein Werkzeug benutzt, um bei die Entscheidung zwischen den zahlreichen auf dem Markt verfügbaren Softwarelösungen zu helfen. Die entsprechende Ontologie wurde aus den Gegebenheiten in der Firma erzeugt, die diese Entscheidung beeinflußen können. Die Eigenschaften der Elemente der Ontologie sollen es erlauben die Bestandteile (wie Webserver, Fähigkeiten, Angestellte) zueinander in Beziehung zu setzen, so dass man am Ende einen logischen Schluss ziehen kann, darüber welche Lösung am besten zu den Voraussetzungen paßt. Die Beziehungen werden in Bildern abgebildet, die die Verknüpfungen der Elemente untereinander zeigen. Doch während einzelne Verbindungen gut analysiert werden konnten, stellten sich größere Netzwerke von Elementen als schwer zu lesen und zu interpretieren heraus.

Die zweite Ontologie behandelt Sun Microsystem's "Change Acceptance Program", ein Programm um die Akzeptanz von neu eingeführter Software, Werkzeugen oder Arbeitsabläufen zu erhöhen. Obwohl das offizielle Handbuch mehrere Werkzeuge (auf Papier) mitbringt, entstand die Idee eine Ontologie als Status Monitor zu verwenden. In ihr kann der aktuelle Stand der vielen zu beachtenden Umstände festgehalten werden (z.B. die Einstellung von Personen gegenüber der Veränderung). Als Ergebnis bleibt festzuhalten, dass eine Ontologie prinzipiell dazu geeignet erscheint, als Status Monitor zu dienen, der Aufwand eine zusätzliche Software zu bedienden und unter Umständen erst zu erlernen gleicht den gewonnen Vorteil jedoch wieder aus, den man erhält wenn man eine Ontologie benutzt.

Conceptual formulation

Original task in German Ontologien sind formal definierte Systeme von Konzepten und Relationen. Darüber hinaus enthalten Ontologien Inferenz- und Integritätsregeln. Gewissermaßen ähnelt eine Ontologie einem UML-Klassen-Diagramm, das Klassen, deren Eigenschaften und die Beziehungen zwischen diesen Klassen modelliert. Ontologien dagegen beziehen sich auf Konzepte, statt auf Softwareklassen.

Bei der Entwicklung von Werkzeugen und Software für die Infrastruktur eines internen Firmennetzwerks (Intranet) müssen viele Parameter berücksichtigt werden. Diese Diplomarbeit soll die Anforderungen, die bei einer solchen Entwicklung auftauchen, systematisch als Konzepte, Relationen, Inferenz- und Integritätsregeln in einer Ontologie formalisieren, so dass für zukünftige Projekte in diesem Feld die Ontologie als Hilfsmittel herangezogen werden kann.

Darüber hinaus soll das "Change Acceptance Program" von Sun Microsystems ebenso in einer Ontologie erfasst werden. Das "Change Acceptance Program" bietet eine Methodologie zur Schaffung von Benutzerakzeptanz bei neuer Software/Infrastruktur. Durch die Erstellung einer Ontologie für diese Methodologie sollen die allgemeinen Konzepte zur Akzeptanzförderung und deren Beziehungen formalisiert und dadurch für ähnliche Projekte bereitgestellt werden.

URL of thesis on university website

- http://www.pst.ifi.lmu.de/DA_Fopra/da-lederer.html

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbstständig angefertigt, alle Zitate als solche kenntlich gemacht sowie alle benutzten Quellen und Hilfsmittel angegeben habe.

München, 31. Mai 2007

. . . .

Acknowledgement

During the creation of this thesis and its report i received a lot of support and counsel from many people. Because i am very grateful for this help, I would like seize the opportunity and thank those persons.

First i would like to express my gratitude to *Prof. Dr. Rolf Hennicker*, for agreeing to a cooperation with Sun Microsystems for the production of this thesis. He was always very open-minded and provided invaluable suggestions and guidance throughout the whole production time.

My co-workers at Sun Microsystems also supported me with a lot of suggestions, and discussing with them led to a lot of ideas. As representatives for all team members i would like to thank my thesis supervisors from Sun Microsystems, *Bernd Domansky* and *Christine Jeske*.

Finally i thank my friends, my family and my wife *Anja*, because they didn't let me down, even though i made myself quite scarce.

Contents

Chapter 1

Introduction

“No longer do we have to go looking for information – we can find it on the Internet, our intranet, or our extranet with the click of a mouse.”

Jim Barksdale, CEO Netscape 1995 – 1999 [JB]

1.1 Motivation

Intranets Jim Barksdale got it right. Information is just a click away. Yet it has to be brought there, and the information has to be made available by mechanisms like grouping, sorting and classification. How much value would a library full of books have, if the desired book can not be found? How much value would web pages have, if the desired information can not be found?

In an intranet, defined as web pages being served to a closed user group only, mostly in an internal company network, (business-related) information is of high value. It will contain information about business partners, important contact data, maybe even non-public research results.

Change information But how does the information get there in the first place? And how do you find it there? An important aspect of information sharing is efficiency of finding desired data. It is not very efficient if a person looking for an information has to contact a colleague, who is likely to have the desired information, and ask him to send the information (or the link to it) over. There are few companies that could work this way, in times of flexible office hours, and even flexible office locations (“roadwarriors”, work from home, etc.).

The intranet has to become an autonomous source of information, where information is easily accessible. And in fast moving times, where information obsolescence and information half-life are crucial problems, the traditional way of maintaining intranet pages is not acceptable anymore.

In this traditional approach a person (often called webmaster/webmistress, webadmin or even "web guru") or a group of such persons existed, that you would send your content or modifications to, who would then take care of these changes. But as a most likely only technical person he/she would sometimes not have the expertise in financial, marketing or HR related things, so the changes would need to be verified by an expert. Altogether this process was/is very time consuming, and therefore not very efficient (see image 1.1).

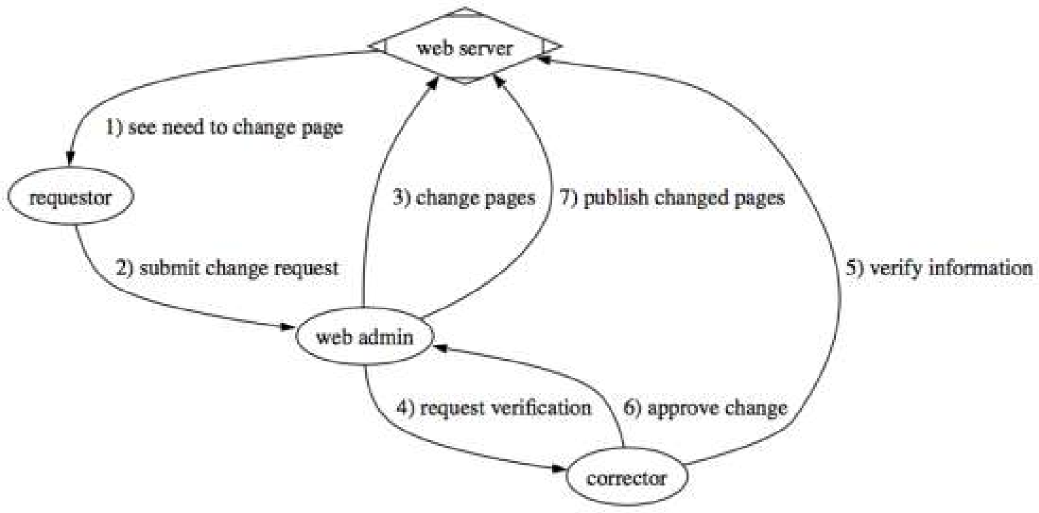


Figure 1.1: *"Traditional", time consuming process of changing information on intranets*

In contrast to the traditional process, that relies heavily on communication between several persons involved, modern systems, that are based upon CMS (content management systems) or wiki (from the Hawaiian term "wiki wiki", meaning quick [Cun]) frameworks, present a way to combine the functions of all persons involved in the traditional process, so that one person can make all the necessary changes. This saves a lot of time, that can then be used to develop business, to acquire new or support existing customers or, in the case of the web administrator, develop or improve existing tools. The modern process can be seen in image 1.2.

It must be mentioned however, that these systems do not allow access to the intranet contents in a "true" wiki way ("full access to everybody"), but rather symbolise the approach of the techniques used to change the actual information. In

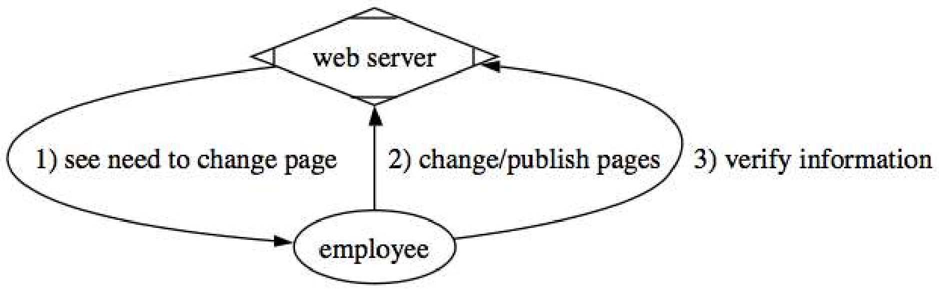


Figure 1.2: *"Modern", fast process of changing information on intranets*

a productive environment a lot of security precautions have to be taken to exactly select who has what sort of access to which information.

There are closed groups for all kind of information (i.e. HR, financial) and not everybody who will be allowed to read a certain information will have write/edit access to it as well. The security-improved approach (with user rights) is actually available in most wiki and CMS systems.

Best Choice So of course it seems to be a good, if not brilliant idea to utilise this sort of technology for the own intranet. But a quick research ([Mat]) will reveal that choosing the right platform is not an easy thing to do. Especially as the decision to migrate to a new form of data storage should not be made frivolously, because of the time and effort needed to actually move the information to the new system.

Ontologies This thesis starts from the idea that any company or department can be described by parameters, and that these parameters can be captured by an ontology. This ontology should then enable a decision maker to choose which platform is the most suitable for his/her company's needs. An ontology is a data model, that represents concepts and relations between them, and as such an ontology allows to decrease the complexity of the system with all its parameters, so that reaching a decision is simplified tremendously.

Change Acceptance The second part of this thesis consists of a different kind of ontology. Every change in intranet or software tools has to be fully accepted to allow users to do work of high quality. To reach a high level of acceptance a lot of convincing is necessary. If there is a new tool around, not only the persons that will use it in the future, but also managers, technical maintainers and others have to be assured of the advantages. This is usually done in a project called “Change Acceptance Program” (or short CAP).

The display of ontologies regarding intranet technology led to the idea, that a project and its related persons could also be captured by an ontology. And this ontology could be used as a tool by the project manager, conducting the CAP, to track and share the current state of the CAP.

1.2 Organisation of this Paper

Basic terms will be defined in chapter 2, which will also introduce the reader to ontologies in common and show an exemplary ontology. *Protégé* is presented, as an ontology editor and visualization tool.

Chapter 3 introduces the intranet ontology, which is one of two main parts of this thesis. The ontology is explained in detail and examined for the objective, which is finding the best possible platform for intranet pages, in regards to the parameter of the network (for example available technology). At the end of this chapter the results of this examination are summarized and evaluated.

The second ontology that was created in the process of this thesis is described in chapter 4. It focuses on Sun Microsystems’ “Change Acceptance Program”, which provides methods and processes to enhance the acceptance of new software or techniques by the people that have to work with them. The ontology that should reflect the CAP’s status is discussed here. At the end of the chapter the results and evaluations are presented.

Finally the results from the examinations of the two ontologies will be summarized and evaluated. An outlook is also presented along with possible improvements to the ontologies. This roundup will be done in chapter 5.

Chapter 2

Ontologies

Introduction After defining the term “ontology” in this section, an example of an ontology will be given. Finally *Protégé* will be introduced: a free, open-source software package, which is an ontology editor and knowledge-base framework from Stanford University. Its purpose is to create, edit, and visualize ontologies [Unib].

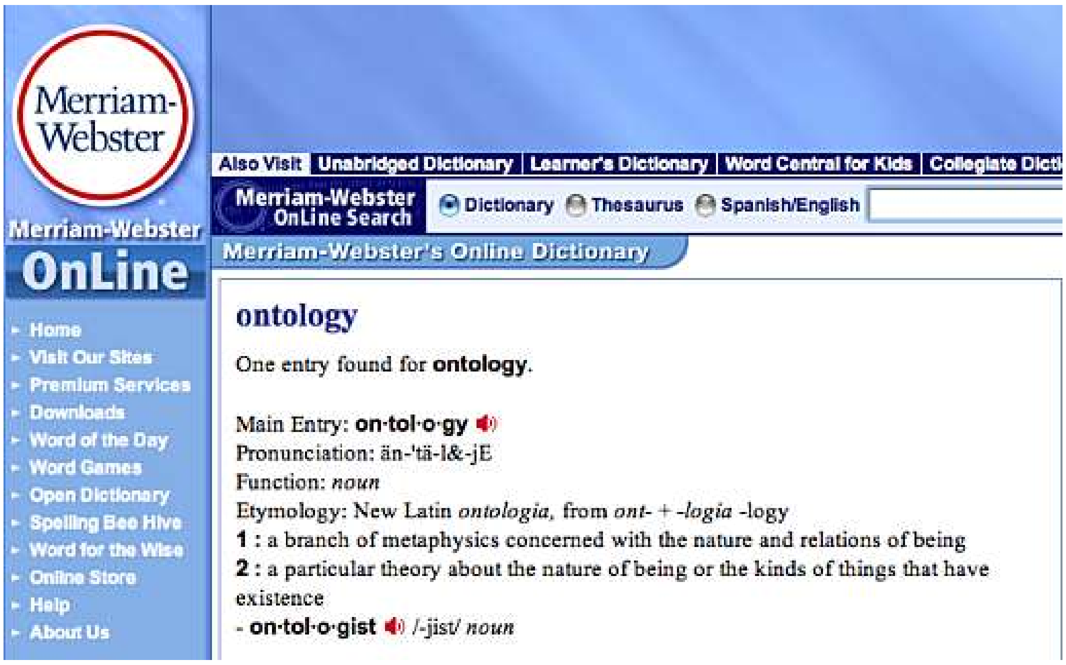
2.1 General Preliminaries

Definition The term “ontology” is used to describe several concepts. Originally a philosophical term, nowadays the word “ontology” is rather connected to information sciences, and describes the methods of information organization. However, looking up the term results in various definitions, of which one shall be quoted here. As you can see in image 2.1 the Merriam-Webster OnLine Dictionary [MW] defines “ontology” as

1. a branch of metaphysics concerned with the nature and relations of being
2. a particular theory about the nature of being or the kinds of things that have existence.

The usage of the term in information and computer science is connected with the meaning of the metaphysical branch, as an ontology is usually also concerned with the relations between its elements, just like the philosophical discipline.

Historical overview It is commonly accepted, that the idea of ontologies was developed by Plato and his student Aristotle in early Greece. The first documented appearance of the word ontology, or rather its Latin form (“ontologia”), can be



ontology

One entry found for **ontology**.

Main Entry: **on·tol·o·gy**
Pronunciation: än-'tä-l&-jE
Function: *noun*
Etymology: New Latin *ontologia*, from *ont-* + *-logia* -logy
1 : a branch of metaphysics concerned with the nature and relations of being
2 : a particular theory about the nature of being or the kinds of things that have existence
- **on·tol·o·gist** /-jist/ *noun*

Figure 2.1: *Merriam-Webster's OnLine Dictionary [MW], defining the term "ontology"*

found in the work "Ogdoas Scholastica" [Lor] by Jacob Lorhard from 1606, and in 1613 in the "Lexicon philosophicum" by Rudolph Göckel [Vii]. Further information about the ancient history of ontologies can be found in several publications like Mika Viinikkala's document "Ontology in Information Systems" [Vii].

Only recently, the term of "(formal) ontology" has been up taken by researchers in information and computer science, where it is used to describe elements of any informational system (see e.g. "What is an ontology?" [Gru]). Thus, an ontology becomes the basic level of a knowledge representation scheme, i.e. a framework.

2.2 Example

This section gives a brief example of an ontology. It is an ontology about pizzas that is used in the OWL tutorial [MH+04].

Pizza example This ontology is used throughout the tutorial guide to demonstrate the creation of an ontology as well as the capabilities of ontologies in general. To clarify the purpose of ontologies, the "Pizza example" is frequently used in the aforementioned tutorial. It shows the usage and structure of an ontology and offers the user an introductory approach to the ontologies. In the tutorial the ontology is created together with the reader, so that the basic procedures are properly understood. A selection of the ontology's properties can be seen in image 2.2, and in image 2.3 the class hierarchy of pizza toppings is shown.

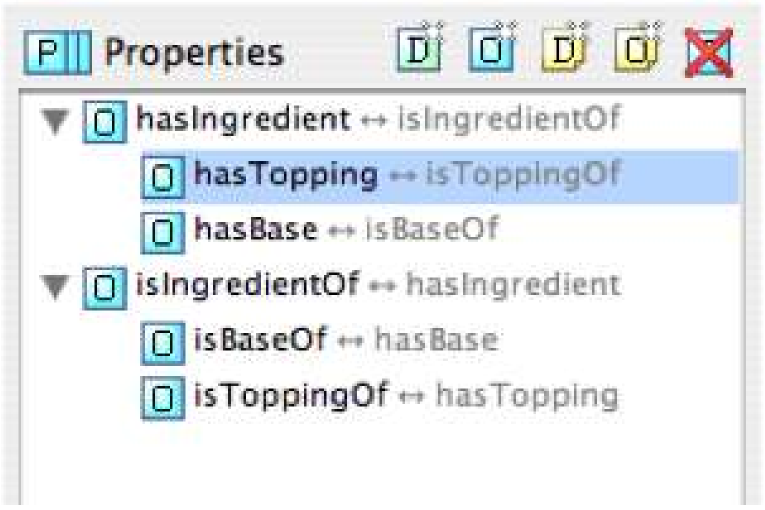


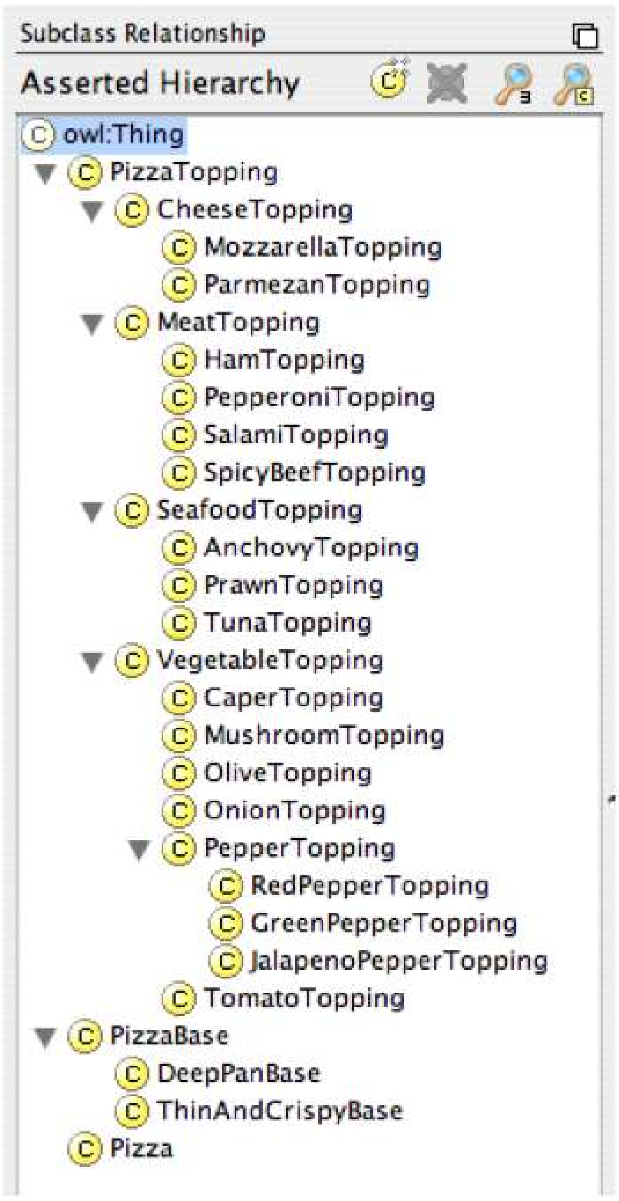
Figure 2.2: *Pizza properties – from Protégé OWL tutorial [MH+04]*

2.3 Conventions

Visualisation Throughout this thesis paper the elements of an ontology will look the same to avoid confusion. The visual representation of classes, individuals and properties will look like the following examples:

- **(Sub-)Class**: **PERSON**, **SERVERSOFTWARE**, **COMPANY**
- **Individual**: ***Solaris***, ***Apache***, ***PHP***, ***Reading***
- **Properties**: *isEmployedBy*, *hasKnowledge*, *usesLanguage*

In images that display ontologies or parts of them, elements will be displayed as shown in image 2.4. Visible in this image is a class: **COMPANY**, an individual: ***Sun_Microsystems*** and properties: ***hasAvailableOS*** and ***hasAvailableWebserver***. Solaris, Linux-Unix and Apache are values of properties (they are also individuals of other classes).



```
owl:Thing
  PizzaTopping
    CheeseTopping
      MozzarellaTopping
      ParmezanTopping
    MeatTopping
      HamTopping
      PepperoniTopping
      SalamiTopping
      SpicyBeefTopping
    SeafoodTopping
      AnchovyTopping
      PrawnTopping
      TunaTopping
    VegetableTopping
      CaperTopping
      MushroomTopping
      OliveTopping
      OnionTopping
      PepperTopping
        RedPepperTopping
        GreenPepperTopping
        JalapenoPepperTopping
      TomatoTopping
  PizzaBase
    DeepPanBase
    ThinAndCrispyBase
  Pizza
```

Figure 2.3: *Pizza class hierarchy – from Protégé OWL tutorial [MH$^+$04]*

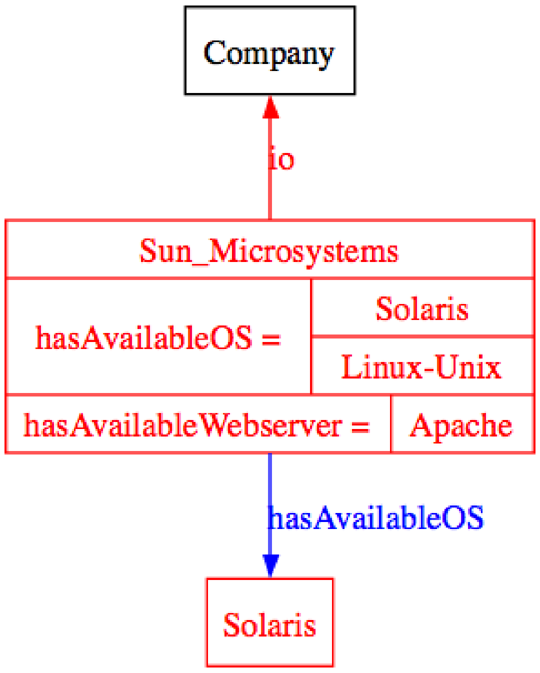


Figure 2.4: *Exemplary display of ontology elements*

Definitions In the context of this thesis, the term ontology will be used to describe a specification of concepts, individuals and relationships that can exist between these concepts and individuals.

What does that mean? In an ontology, concepts and representatives of these concepts will be noted down, brought together, and maybe linked together, depending on the relationship between the items. This way you can see which items are closer to each other, and have more connections to commonly linked concepts.

In accordance with current scientific usage the following terms will be used throughout this thesis:

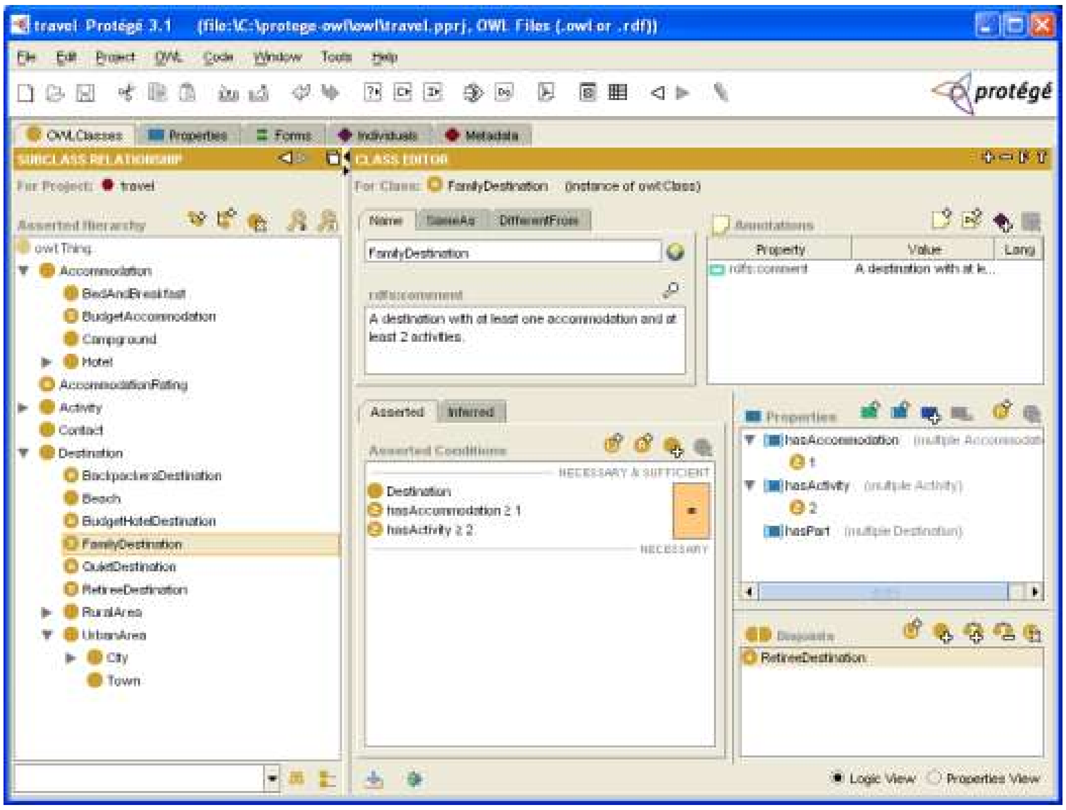
- **(Sub-)Classes** describe concepts or subconcepts. They are the main elements of an ontology.
- **Individuals** are the unique, real-world instances of concepts.
- **Properties** exist in two different kinds. Object properties connect these aforementioned classes and individuals with each other and thus represent a relationship between them (**MR. SMITH** *owns* ***Mr. Smith's car***.). Datatype properties simply add information to an item (**MR. SMITH** *has age* 42 years) just like a variable containing a certain value.

An in-depth view on the elements of an ontology is available in the "Practical Guide To Building OWL Ontologies" [MH+04], Chapter 3.

2.4 *Protégé* – a tool to create and edit ontologies

In this subsection *Protégé* will be introduced and some of its views are presented. The images (with the exception of 2.8) are taken from the *Protégé* website [Unib].

Protégé is a platform-independent software to create and edit ontologies. In image 2.5 the class editor window can be seen. It is used to configure what defines a class. In the right hand area the field with the class properties can be seen. Here the class **FAMILYDESTINATION** is detailed, and it has the properties *hasAccomodation*, *hasActivity* and *hasPart*.



A destination with at least one accommodation and at least 2 activities.

Figure 2.5: Protégé class editor window

Individuals, real-world examples of classes, are edited in the window shown in image 2.6. Here you can see the individual ***Sydney*** on display. To clarify the terms: In this image you see the individual ***Sydney***, which is an example of the

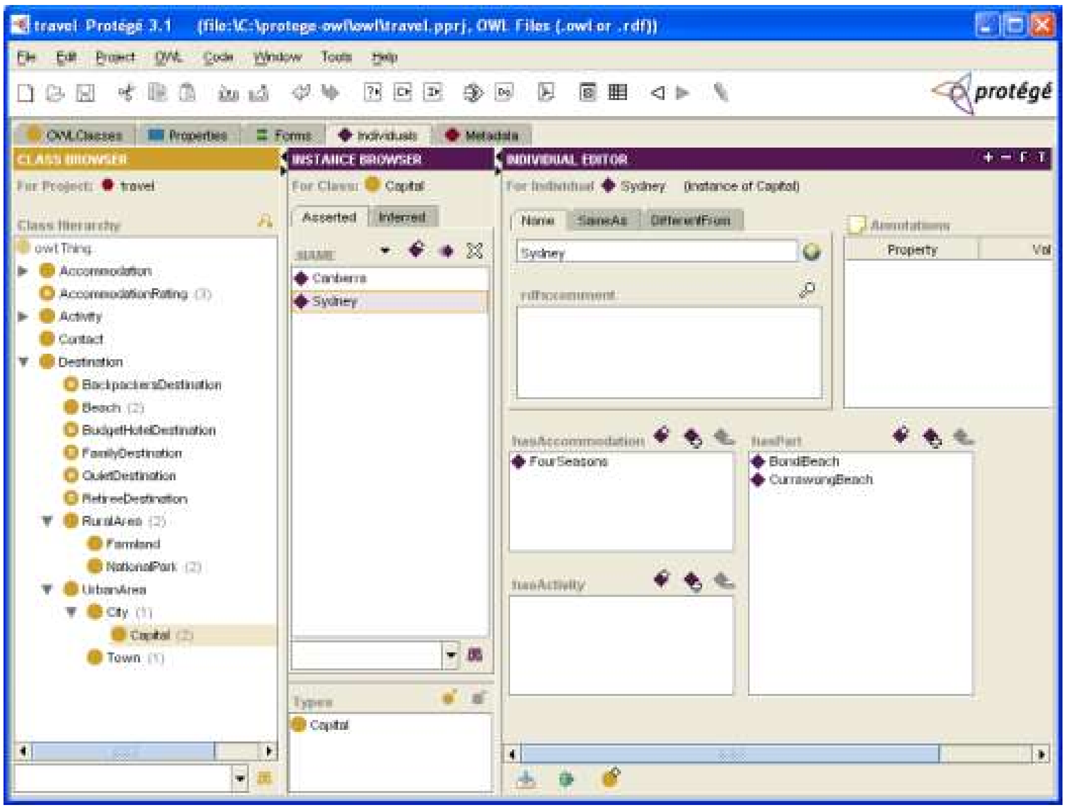


Figure 2.6: Protégé individual editor window

subclass **CAPITAL**, which in turn is the leaf of the class tree in the succession of the (sub-)classes: **CITY**, **URBANAREA**, **DESTINATION** and the all-embracing **OWL:THING** (left part of the windown: CLASS BROWSER). On the right side the properties of the individual are editable: *hasActivity*, *hasAccomodation* and *hasPart* (right part of the window: INDIVIDUAL EDITOR). Individuals can belong to more than one class, like in this example, a destination ***BeachCity*** could as well be a **CITY** and a **BEACH** destination.

In image 2.7 the properties editor is shown, displaying the *hasAccomodation* property. In the main area you can define domain and range of a property, and you have the option to define the property as functional and to define its reverse (if existing). Domain and range are the elements that can be linked with this property. In the example detailed below the property *hasAccomodation* links **DESTINATION** (domain) to **ACCOMODATION** (range).

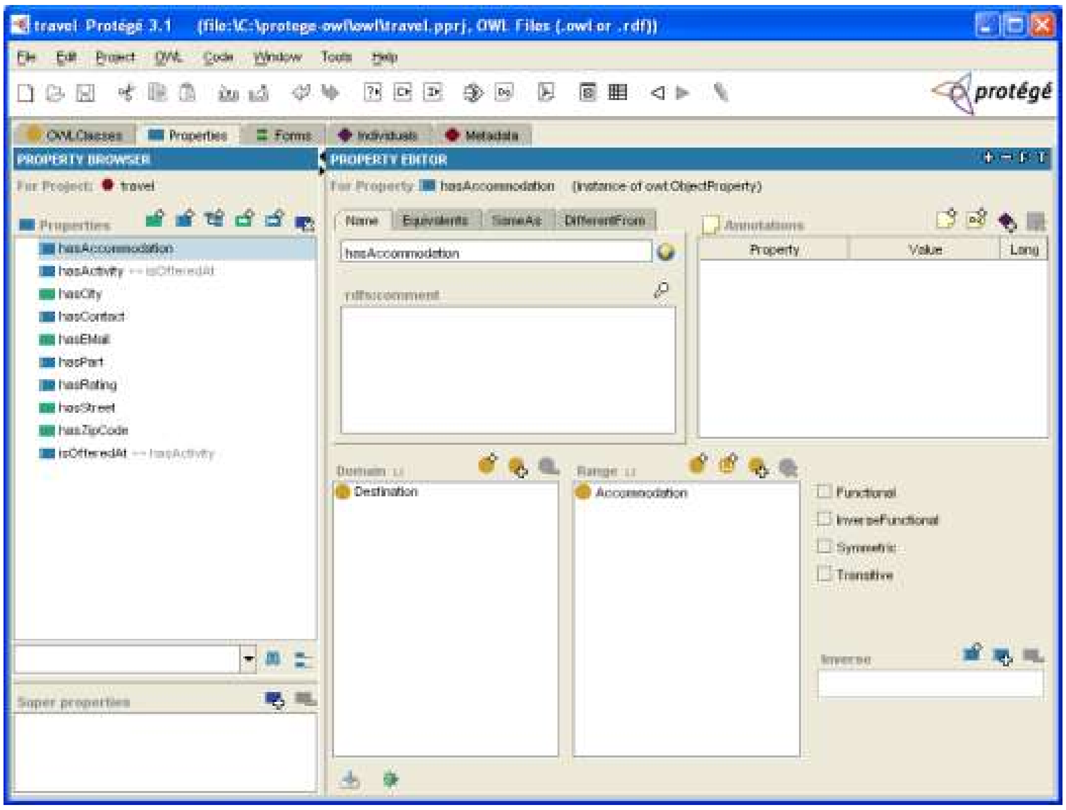


Figure 2.7: Protégé properties editor window

In the last image (2.8) the visualization Ontoviz [Unia] plug-in for *Protégé* can be seen. This image shows the relationships between several classes and subclasses. But the Ontoviz plug-in has a lot features, allowing a variety of depictions of data from within an ontology. All images in the later sections of this thesis paper, detailing the produced ontologies of this thesis, are generated with the Ontoviz plug-in.

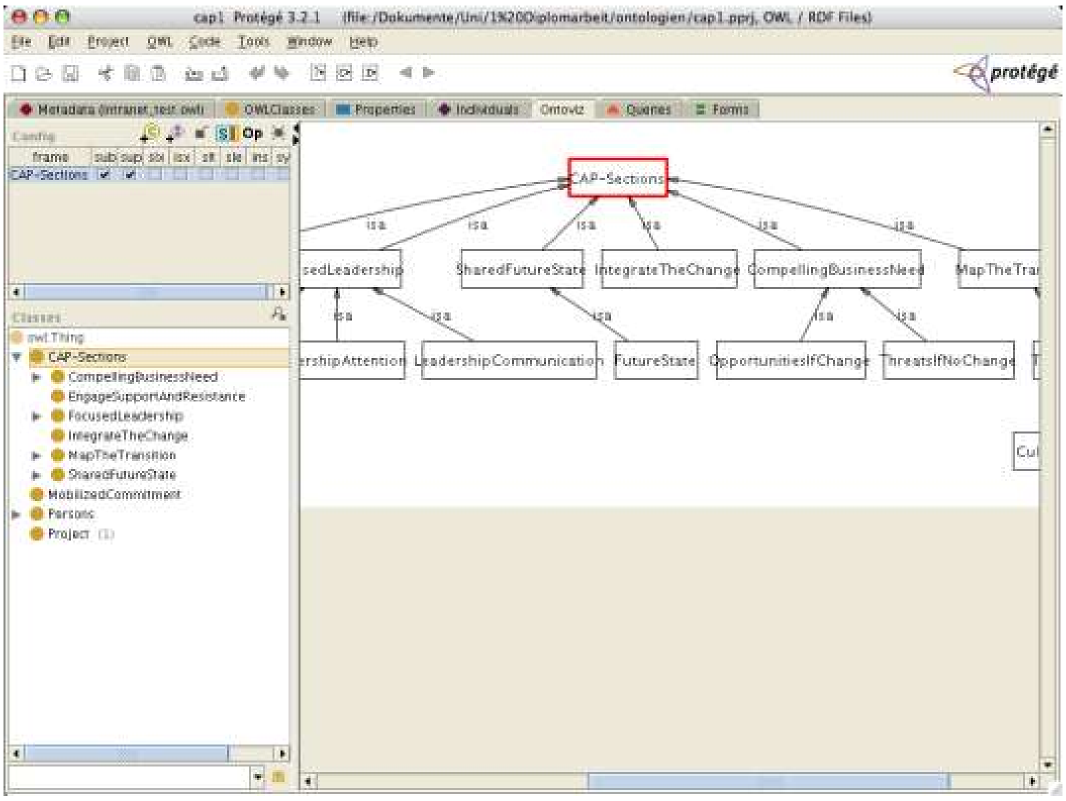


Figure 2.8: Ontology visualisation with Ontoviz Plugin

Chapter 3

Intranet Ontology

Introduction Every change is started by the need or want to improve. In this section the change of the underlying technology of an intranet will be examined. The motivation to do this, and why this is approached with an ontology is described in the first section. Thereafter the details of the ontology are given, and the section is concluded by the result of evaluating possible platforms for an intranet infrastructure through the use of the ontology.

3.1 Motivation

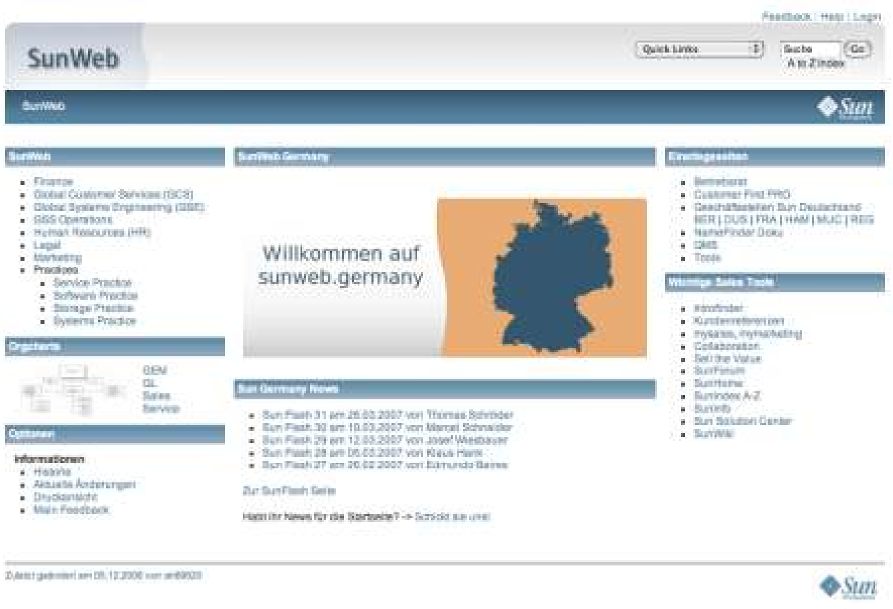
Background Intranet pages have become a source of important information for those that can access it. In it not only in-depth information about products (more detailed as in the company's external accessible internet pages) can be found, but also reference documents, solutions and pricing information are available for employees and partners of a company. Information that does not have to be displayed with a fancy layout or latest technological gizmos, but rather has to be available quick, clean and reliable.

In Sun Microsystems' Global Sales and Services Organization internal information is used for various tasks. Aside from tools that allow employees to reserve workspace (desks) or applications that offer address book information of colleagues, simple information transfer from one employee to another is the primary assignment of the intranet. To name only a few, the following are among the daily tasks that are performed by Sun employees with the help of the intranet:

- offer various levels of support services to calling customers and clients by looking up support information about hardware and software

- composing offers to potential customers
- share experiences and lessons learned from past projects
- archive finished projects
- store meeting agendas and minutes

For these (and other) tasks Sun Microsystems offers intranet servers to its employees, which usually refer to the system by the term “SunWeb” [Sun]. Each country has its own starting point, which is accessible in the internal network at the URL http://sunweb.*country*. So the German pages are located at http://sunweb.germany. (see figure 3.1)



Feedback | Help | Login

SunWeb

Quick Links | Suche | Go | A to Z Index

SunWeb

SunWeb
- Finance
- Global Customer Services (GCS)
- Global Systems Engineering (GSE)
- GSS Operations
- Human Resources (HR)
- Legal
- Marketing
- Practices
  - Service Practice
  - Software Practice
  - Storage Practice
  - Systems Practice

Orgcharts

OEM
GL
Sales
Service

Optionen

Informationen
- Historie
- Aktuelle Änderungen
- Druckansicht
- Mein Feedback

SunWeb Germany

Willkommen auf sunweb.germany

Sun Germany News
- Sun Flash 31 am 26.03.2007 von Thomas Schröder
- Sun Flash 30 am 19.03.2007 von Marcel Schneider
- Sun Flash 29 am 12.03.2007 von Josef Westbauer
- Sun Flash 28 am 05.03.2007 von Klaus Hardt
- Sun Flash 27 am 26.02.2007 von Edmundo Baires

Zur SunFlash Seite

Habt ihr News für die Startseite? -> Schickt sie uns!

Einstiegsseiten
- Betriebsrat
- Customer First PRO
- Geschäftsstellen Sun Deutschland
  BER | DUS | FRA | HAM | MUC | REG
- NameFinder Doku
- QMS
- Tools

Wichtige Sales Tools
- Infofinder
- Kundenreferenzen
- mySales, myMarketing
- Collaboration
- Sell the Value
- SunForum
- SunHome
- SunIndex A-Z
- SunInfo
- Sun Solution Center
- SunWiki

Zuletzt geändert am 05.12.2006 von [illegible]

Figure 3.1: *Starting page of Sun Microsystems’ German intranet [Sun]*

Intranet at Sun Microsystems To understand the need to modernize the way information is stored and actualized in the internal net, and the need to base the new solution on a platform that is capable of adapting to future needs, one has to understand the status of Sun’s intranet at the point in time, when this thesis was started. At first there was no central web server in Germany. Several divisions offered to host internal web pages.

But even later, after a main web server had been installed, several departments chose to set up their own hardware for the following reasons:

- being able to install new, custom software
- serving content from a seemingly more available system, due to network issues
- allowing several persons to maintain the web pages
- eliminating the restraint to maintain information with a predetermined technology
- trimming the process of request changes from an designated person (webmaster)

Allowing departments to have their own web servers and to use their favorite techniques to produce and maintain the contents of their web pages, also resulted in several different layouts over the time, as the corporate design (which is part of the corporate identity) was changed several times over the past years. And as not all intranet pages have been taken care of with the necessary time and effort to adapt the newer layouts at some point of time, the whole intranet presented itself with several different layouts. In addition to that, the intranet consisted more or less of pure static HTML pages which combined content and layout in one file (an approach commonly avoided nowadays).

While the underlying techniques are not a hindrance to the user (reader) of pages, the layout and navigation structure and therefore the usability of the pages resulted in less efficient information procurement of the reader. In addition to that the effort necessary to keep pages (static HTML) up to date were a challenge to the maintainers.

Also notable is the circumstance that the organisation of the company has been restructured several times in the past years as well: subdivisions were shifted in reporting lines and managers changed or were replaced. This generated two effects: information was also relocated to reflect the new organizational structures (but sometimes copied and not moved, rendering the original unmaintained and sooner or later inaccurate), and new leaders had a different opinion on internal web pages, and so web pages were restructured, renewed and redone quite often.

Altogether this led to several disadvantages:

- decreased usability due to different control structures and navigational elements
- increased and repeated costs to generate and update pages
- broken links between connected content due to move
- missing links because content is not advertised
- unmaintained content due to change of personnel or change of departmental structures
- lack of a central starting point for information search

All pages that users could access were maintained by different users with different tools or liking. Even though there was an official Sun tool to generate and publish intranet pages, called “sunpublish”, this was often not used by users for one of the following reasons:

- users planning to put up a few pages only, not wanting to learn an additional tool
- persons inherited the maintenance of web pages due to structural changes in the company
- persons lacked the necessary experience with web technology

This resulted in a lot of different approaches to user guidance in the interface and navigation. A comparison of the navigational elements and their placement can be drawn on the images 3.2, 3.3 (outdated layouts) and 3.1 (current layout).

Redesign, migration, improvement The aforementioned central SunWeb server in Germany is administered by the Technical Operations department of Sun Microsystems. This team obtained the task to consolidate the different intranet pages and to make them accessible by users more easily. It was decided that a new approach to web page maintenance had to be made in the light of the central importance of a company’s intranet, and so as part of this thesis all those disadvantages had been detected, and a way to remove and avoid such disadvantages in the future was searched.

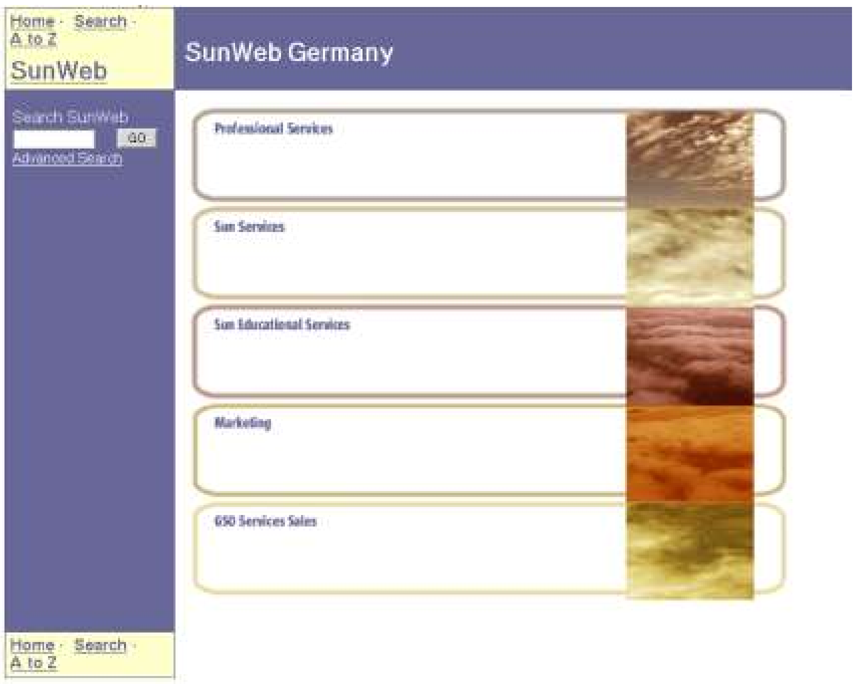


Figure 3.2: *Outdated layout of Sun Microsystems' German intranet 1 [Sun]*

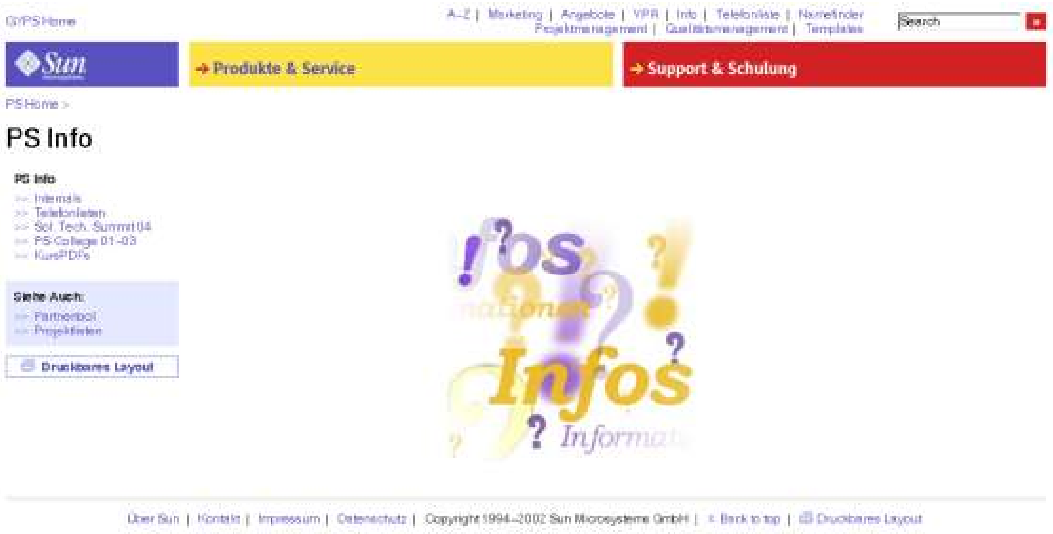

GYPS Home
A–Z | Marketing | Angebote | VPR | Info | Telefonliste | Namefinder
Projektmanagement | Qualitätsmanagement | Templates
Search
Sun
Produkte & Service
Support & Schulung
PS Home >
PS Info
PS Info
>> Internals
>> Telefonlisten
>> Sol. Tech. Summit 04
>> PS-College 01–03
>> KursPDFs
Siehe Auch:
>> Partnertool
>> Projektlisten
Druckbares Layout
Infos
Über Sun | Kontakt | Impressum | Datenschutz | Copyright 1994–2002 Sun Microsystems GmbH | Back to top | Druckbares Layout

Figure 3.3: *Outdated layout of Sun Microsystems' German intranet 2 [Sun]*

Why use an ontology? As there are many valid approaches to the problems mentioned above (e.g. CMS, Wiki, databases, PMS[1]), what is the best way to compare these systems and to decide on the one to use? With the use of an ontology it becomes possible to visualize the parameters of the conditions at Sun, the attributes of each platform and all influencing factors, so that in the end it becomes visually obvious which solutions fulfill the requirements best.

In addition to this decision a way to reproduce this process was desirable in order to save time and manpower at similar tasks and problems. So the general idea was to display a system (with all its elements and constraints) and to permit the conclusion, which solution fits best.

General requirements The following are some of the more important concepts and technologies that are part of the intranet infrastructure at Sun Microsystems. The first approach in finding "the best" intranet platform was to look which solution fulfilled the requirement. The system that was to be used as new underlying technology had to be able to use those concepts out of the box or allow a simple implementation of a solution to integrate them.

- Variable user rights. To use the intranet for all kind of information, there has to be an option to restrict read/write access to certain information.
- LDAP access. In Sun Microsystems intranet, users authenticate themselves via LDAP servers. The new platform had to be able to utilise them.
- Flexible Layout. As detailed above page design is changed from time to time. To be able to adopt those changes easily, a flexible template system has to be available.

But it was soon realized that most of the criteria listed above can be accomplished by most modern CMS and wiki systems [Mat, Cos]. Static page systems or publishing systems are not suited for this kind of infrastructure that well. But with so many different possible solutions other characteristics had to be taken into account for a decision. For instance available skills of employees or available operating systems. By the use of an ontology such evaluations should become possible.

[1]Publishing Management System: a software used primarily to publish web pages, regardless of their way of creation. Layout and navgiational elements are added in the publishing process and are therefore congenerous

3.2 Details of the Ontology

As explained earlier, an ontology basically consists of classes, individuals and properties. The elements of the ontology about the intranet will be detailed in this section.

3.2.1 COMPANY class

The **COMPANY** class represents the company that is detailed in the ontology. By the use of it's properties (*hasAvailableOS* and *hasAvailableWebserver*) available operating systems and web servers can be stored with the individual for each company. These values are then used to connect software to a company, to be able to narrow down possible solutions. Also the class (and its individuals) works as node to connect persons to the ontology, which are employed by companies. The **COMPANY** class is shown in image 3.4.

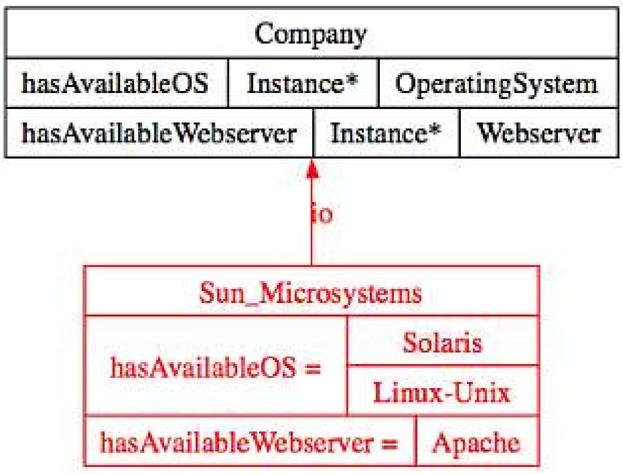


Figure 3.4: **COMPANY** *class with individual* ***Sun Microsystems***

3.2.2 PEOPLE class

In this class persons are categorized into subclasses according to their function within the company. It is depicted in image 3.5. People in this ontology are generally classified by possessing knowledge, which are part of the **KNOWLEDGE** class (see below). Possessing knowledge qualifies them for being an element of certain subclasses.

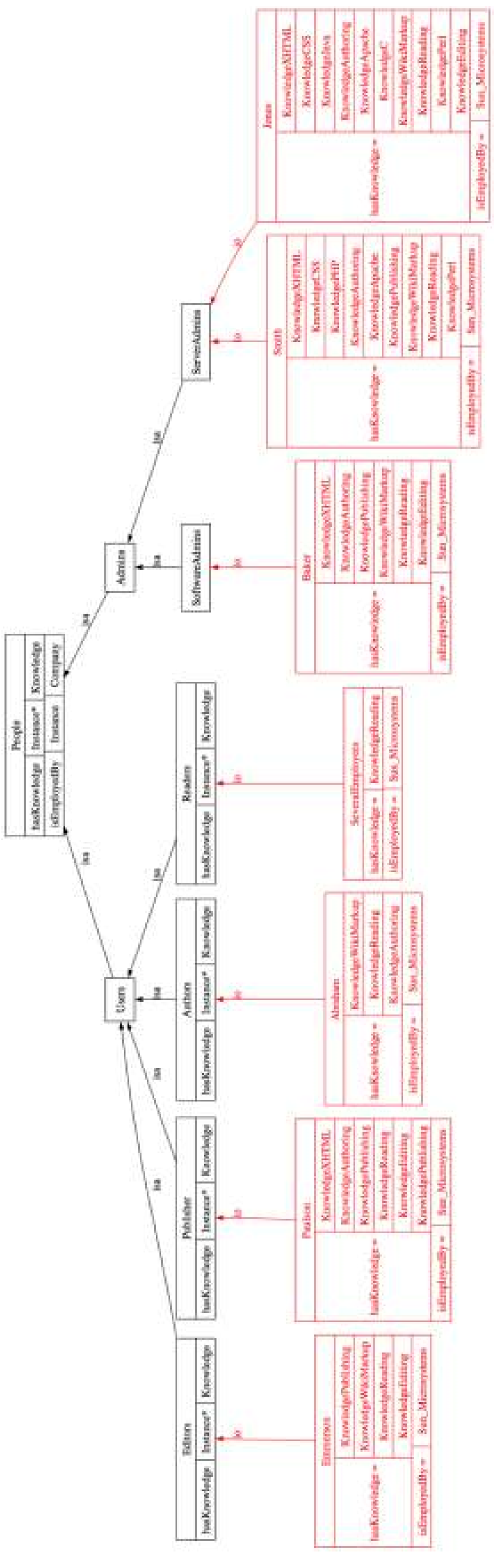


Figure 3.5: **PEOPLE** *class with subclasses and exemplary individuals and properties*

ADMINS To be part of the **ADMINS** class a person has to have a knowledge of the **WEBSERVER** class, which would make them part of the **SERVERADMINS** subclass, or, a knowledge of the **MARKUP** or the **PROGRAMMING** class, which would put them into the **SOFTWAREADMINS** subclass.

USERS As with **ADMINS**, persons holding a special knowledge are divided into several subclasses within the **USERS** class. To become part of the **USERS** class people need to possess **KNOWLEDGE** of **PROCEDURES**, which are needed to operate internet technologies. Aside from those (***Authoring***, ***Editing*** and ***Publishing***) also simple tasks, like ***Reading*** are listed, to be able to categorize Users into meaningful subclasses.

According to the individuals of the **KNOWLEDGE** class given above, the subclasses of **USERS** are **AUTHORS**, **EDITORS**, **PUBLISHER** and **READERS**.

3.2.3 KNOWLEDGE class

The class **KNOWLEDGE** (see image 3.6) represents knowledge a person can have. By allocating knowledge to users, it becomes possible to test against the availability of certain knowledge in a company, to see which platforms can be used and operated. (e.g. if one of the possible intranet platforms requires knowledge of the PHP language, then there should be at least one person possessing knowledge of it who is available to administer the platform.)

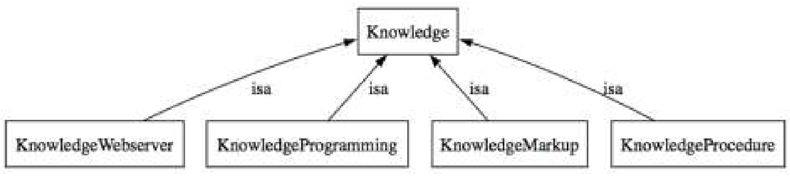


Figure 3.6: **KNOWLEDGE** *class with subclasses*

KNOWLEDGEMARKUP In the **KNOWLEDGEMARKUP** subclass (image 3.7) all skills regarding markup languages are stored. In the exemplary ontology these are ***KnowledgeCSS***, ***KnowledgeXHTML*** and ***KnowledgeWikiMarkup***. These individuals symbolize knowledge of individuals of the **SERVERSOFTWARE** class.

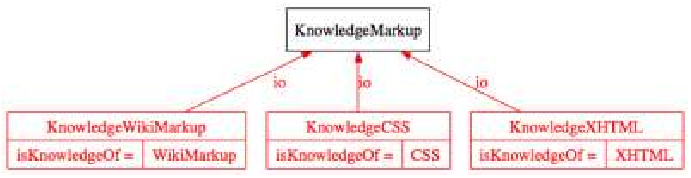


Figure 3.7: **KNOWLEDGEMARKUP** *class with individuals*

KNOWLEDGEPROCEDURE Procedural knowledge is required to operate the information platform on different levels. The individuals of this class are the corresponding knowledge fields of the individuals of the **PROCEDURES** class. As you can see in image 3.8 there are four individuals here: ***KnowledgeAuthoring***, ***KnowledgeEditing***, ***KnowledgePublishing*** and ***KnowledgeReading***. They are used to describe knowledge of the individuals ***Authoring***, ***Editing***, ***Publishing*** and ***Reading*** of the **PROCEDURE** class mentioned below.

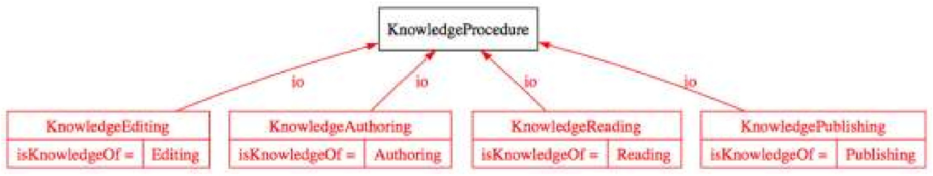


Figure 3.8: **KNOWLEDGEPROCEDURE** *class with individuals*

KNOWLEDGEPROGRAMMING In this class the skills can be found that deal with programming tasks, and therefore are important to members of **SOFTWAREADMINS**. Examples of such skills are ***ASP***, ***C***, ***Java***, ***Perl*** and ***PHP*** which are all individuals of the **LANGUAGES** subclass (see below).

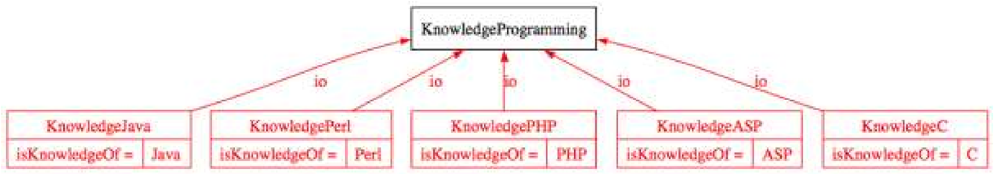


Figure 3.9: **KNOWLEDGEPROGRAMMING** *class with individuals*

KNOWLEDGEWEBSERVER A fourth subclass of the **KNOWLEDGE** class is the **KNOWLEDGEWEBSERVER** subclass. It contains knowledge of web servers like

Apache and ***IIS***.

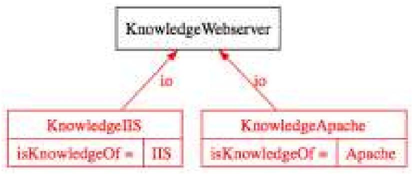


Figure 3.10: **KNOWLEDGEWEBSERVER** *class with individuals*

3.2.4 PROCEDURE class

The individuals of this class represent the procedures that are necessary to operate the different forms of platforms for an intranet. For common HTML editing and viewing only ***Authoring***, ***Editing*** and ***Reading*** might be necessary. ***Publishing*** could become important if system of content management is used where content has to be reviewed before being made generally accessible.

3.2.5 SERVERSOFTWARE class

The software that is operated on a server includes the possible platforms. Therefore this class (**SERVERSOFTWARE**) is an important one, as it contains the possible solutions. In this class all the different aspects of software that is necessary for a web server, as needed to serve pages to an intranet, are available. Most of the individuals in the subclasses are connected to individuals of the **KNOWLEDGE** class, so that persons can be connected to them. A scheme of this class's elements can be seen in image 3.11.

An example of a common individual is shown in image 3.12. Aside from all individuals of the **LANGUAGE** subclass in the middle of the screenshot, the properties of the individual ***Java*** can be seen on the right. The property *canRunOn* is filled with ***Windows***, ***Linux*** and ***Solaris***, all individuals of the **WEBSERVER** subclass. They represent the operating systems that ***Java*** can be operated on.

CONTENTSYSTEM While most of the other **SERVERSOFTWARE** subclasses and individuals are rather straight forward and require no detailed examination, the **CONTENTSYSTEM** is an exception, as it contains the platforms that intranet solutions can be based upon.

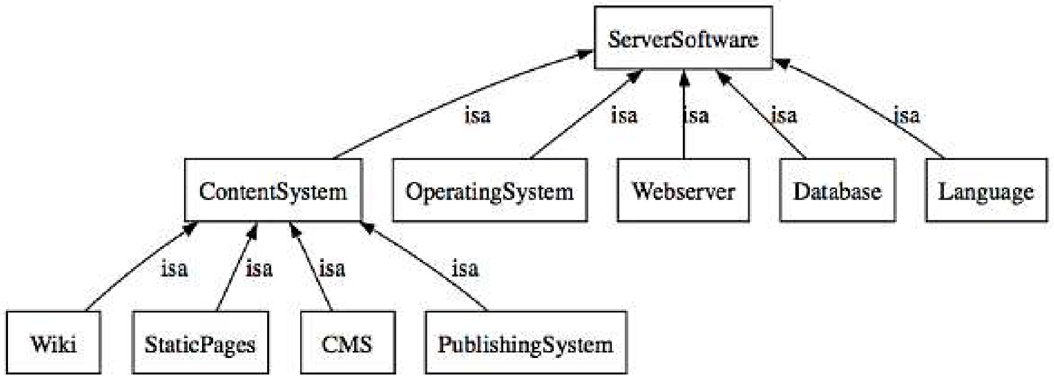


Figure 3.11: *Overview of* **SERVERSOFTWARE** *class*

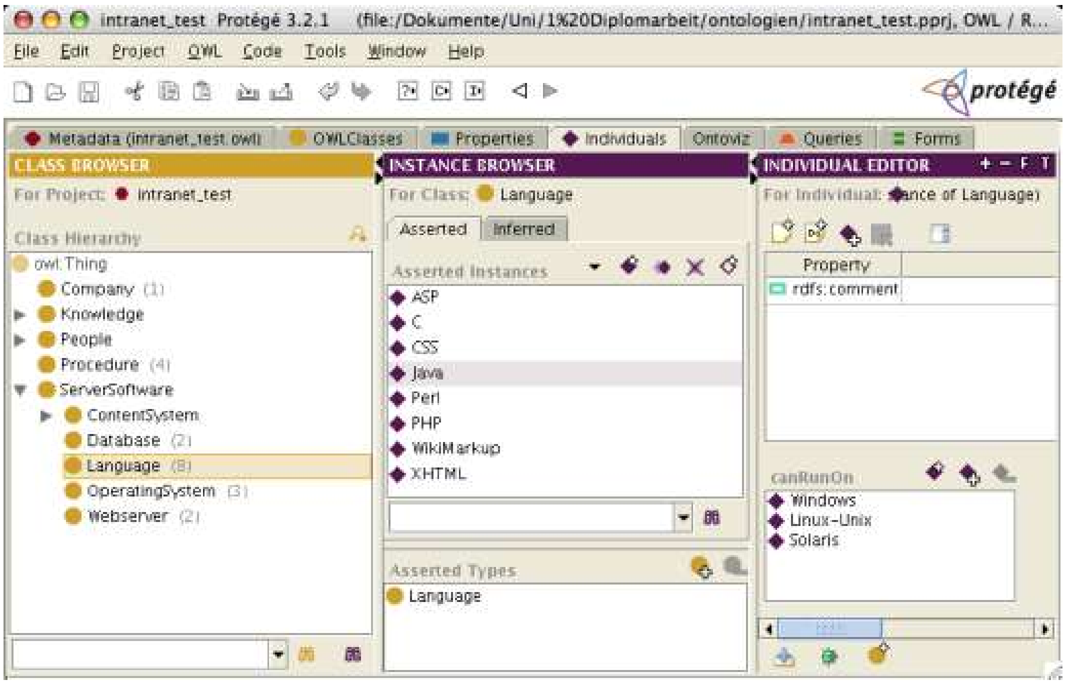


Figure 3.12: Java*, individual of* **LANGUAGE** *class in Protégé editing view*

As mentioned above, this class deserves a closer look, as it contains the answer to the question, that is most likely the most important to the user of the ontology: "Which platform is suited best?" In image 3.13 the individuals of this class together with its properties can be seen. In detail there are:

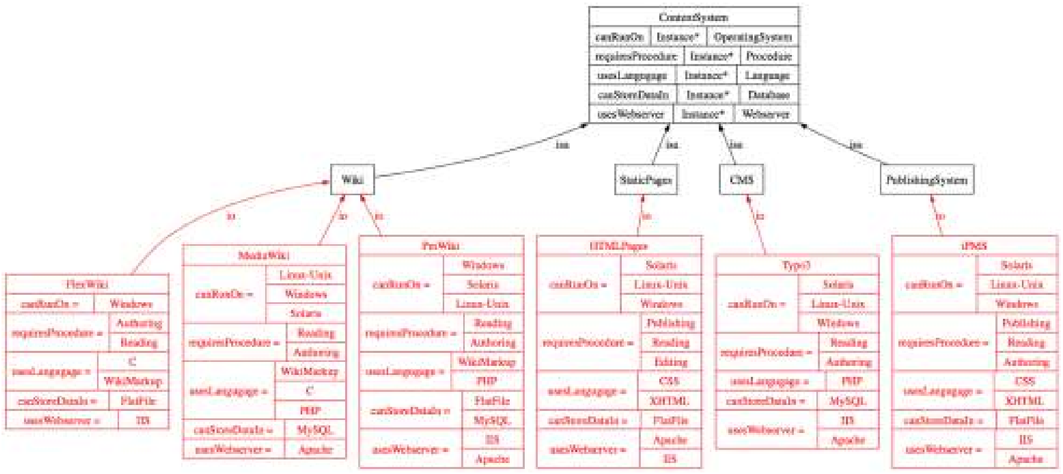

ContentSystem
canRunOn | Instance* | OperatingSystem
requiresProcedure | Instance* | Procedure
usesLanguage | Instance* | Language
canStoreDetails | Instance* | Database
usesWebserver | Instance* | Webserver
isa
Wiki
StaticPages
CMS
PublishingSystem
io
FlexWiki
canRunOn = | Windows
requiresProcedure = | Authoring | Reading
usesLanguage = | C | WikiMarkup
canStoreDetails = | FlatFile
usesWebserver = | IIS
MediaWiki
canRunOn = | Linux-Unix | Windows | Solaris
requiresProcedure = | Reading | Authoring
usesLanguage = | WikiMarkup | C | PHP
canStoreDetails = | MySQL
usesWebserver = | Apache
PmWiki
canRunOn = | Windows | Solaris | Linux-Unix
requiresProcedure = | Reading | Authoring
usesLanguage = | WikiMarkup | PHP
canStoreDetails = | FlatFile | MySQL
usesWebserver = | IIS | Apache
HTMLPages
canRunOn = | Solaris | Linux-Unix | Windows
requiresProcedure = | Publishing | Reading | Editing
usesLanguage = | CSS | XHTML
canStoreDetails = | FlatFile
usesWebserver = | Apache | IIS
Typo3
canRunOn = | Solaris | Linux-Unix | Windows
requiresProcedure = | Reading | Authoring
usesLanguage = | PHP
canStoreDetails = | MySQL
usesWebserver = | IIS | Apache
iPMS
canRunOn = | Solaris | Linux-Unix | Windows
requiresProcedure = | Publishing | Reading | Authoring
usesLanguage = | CSS | XHTML
canStoreDetails = | FlatFile
usesWebserver = | IIS | Apache

Figure 3.13: **CONTENTSYSTEM** *class with individuals and their properties*

WIKI systems Three wikis are in this ontology as examples: ***FlexWiki***, ***MediaWiki*** and ***PmWiki***. Basically they differ in the programming language used and the web server and operating system they can run on. See image 3.14.

STATICPAGES system This class describes the approach of writing HTML pages directly with a text editor or a HTML editor, and storing them onto the server. There is no dynamic in such pages, and even though an basic layout can be separated from the content with the usage of cascading style sheets (CSS) the structure of the page can not be changed to match future redesigns.

CMS system As a typical example of an CMS System, Typo3 is shown as individual of the ontology. While the administrative options to a Typo3 system are very powerful, a database is needed to store the pages. Image 3.16 shows the **CMS** class with the ***Typo3*** individual and its properties.

PUBLISHINGSYSTEM In order to allow pages to be moved to a new layout with ease, and to assure the same look and feel on all pages, Sun provided an

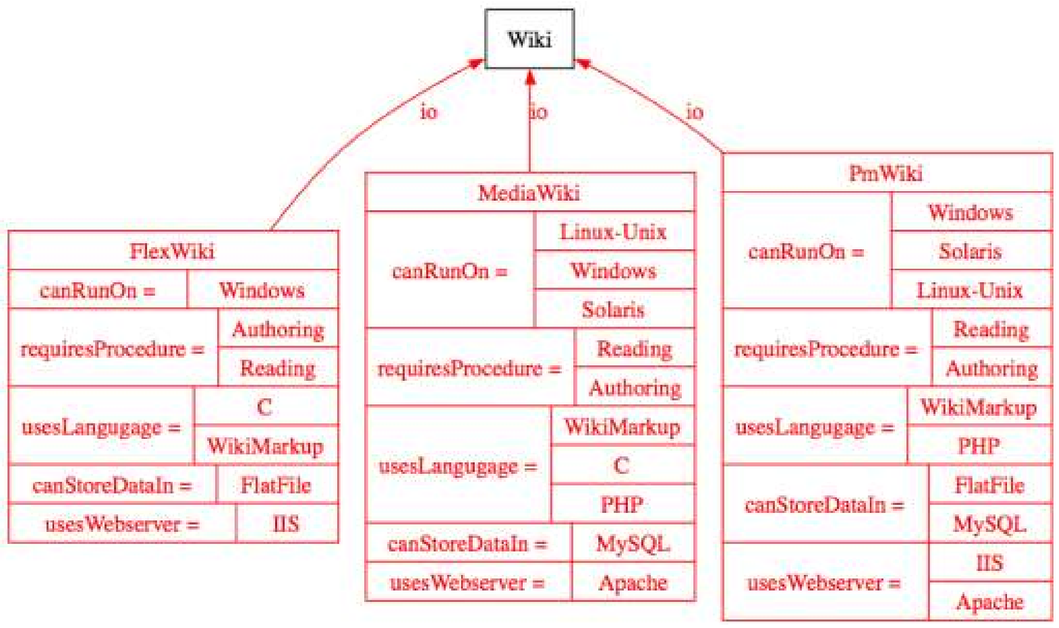

Wiki
io
io
io
FlexWiki
canRunOn = Windows
requiresProcedure = Authoring Reading
usesLangugage = C WikiMarkup
canStoreDataIn = FlatFile
usesWebserver = IIS
MediaWiki
canRunOn = Linux-Unix Windows Solaris
requiresProcedure = Reading Authoring
usesLangugage = WikiMarkup C PHP
canStoreDataIn = MySQL
usesWebserver = Apache
PmWiki
canRunOn = Windows Solaris Linux-Unix
requiresProcedure = Reading Authoring
usesLangugage = WikiMarkup PHP
canStoreDataIn = FlatFile MySQL
usesWebserver = IIS Apache

Figure 3.14: **WIKI** *class with individuals and their properties*

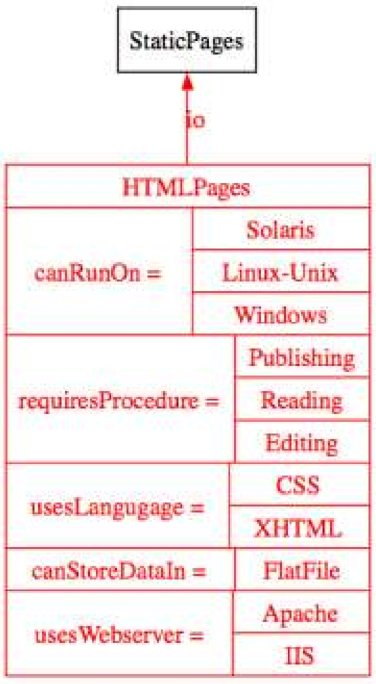

StaticPages
io
HTMLPages
canRunOn = Solaris Linux-Unix Windows
requiresProcedure = Publishing Reading Editing
usesLangugage = CSS XHTML
canStoreDataIn = FlatFile
usesWebserver = Apache IIS

Figure 3.15: **STATICPAGES** *class with individual and its properties*

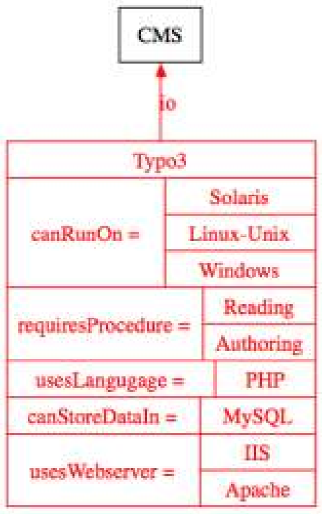


Figure 3.16: **CMS** *class with* ***Typo3*** *individual and its properties*

tool called "internet publishing managament System" (iPMS), where the content of a page had to be written in HTML and deposited in a folder on the server, that corresponded to the URL. At a certain point of time (it can also happen user triggered), a superuser would start the publishing process, and all pages would be built. The building would consist of parsing the folder structure, generating the URL (for links) and adding the navigational elements (in context to the page's location). The pages would then be stored for review in a staging area. After the review the pages were copied to the productive system.

While this was an improvement to simple HTML pages, as the layout was now consistent within all pages, this had several drawbacks too. The most important disadvantage of this process is (as detailed in image 1.1 in chapter 1) the time the whole process needs from requesting the change to having the new content available. In image 3.17 the ***iPMS*** system's properties can be seen.

3.3 Results & Evaluation

3.3.1 Introduction

After explaining the elements of the ontology on detail, the process of deciding on a platform is laid out in this section. As explained earlier in the process of this thesis the Ontoviz Plugin [Unia] is used to generate images of the ontology. The images shown in this section are interpreted and so the quality of a possible solution can

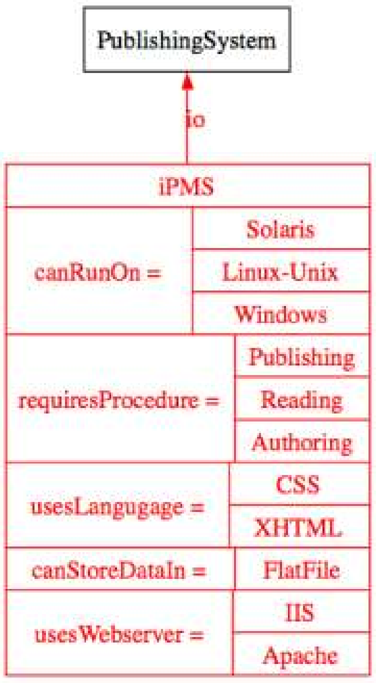
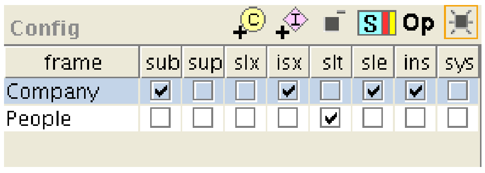


Figure 3.17: **PUBLISHINGSYSTEM** *class with individual and its properties*

be evaluated. Ontoviz allows to set certain parameters in a small interface. These options can be set individually for each displayed element. They are [Unia]:

- sub – subclass closure: subclasses are displayed
- sup – superclass closure: the superclass is displayed
- slx – slot extension: elements, the item is connected to are displayed
- isx – inverse slot extension: elements that are connected to this item are displayed
- slt – slots: properties are displayed in a table at each element
- sle – slot edges: properties are displayed as connections (arrows) between elements
- ins – instances: individuals are added to the image
- sys – system frames: system elements are displayed

The interface can be seen in image 3.18, and the resulting graphic is displayed in image 3.19. Elements of the ontology are added to the graphics as needed (i.e. if a relationship starts at an element already included and ends at a new element) if not added manually to the list in the interface.



frame	sub	sup	slx	isx	slt	sle	ins	sys
Company	✓	☐	☐	✓	☐	✓	✓	☐
People	☐	☐	☐	☐	✓	☐	☐	☐

Figure 3.18: *Ontoviz plug-in options to configure the display of the ontology*

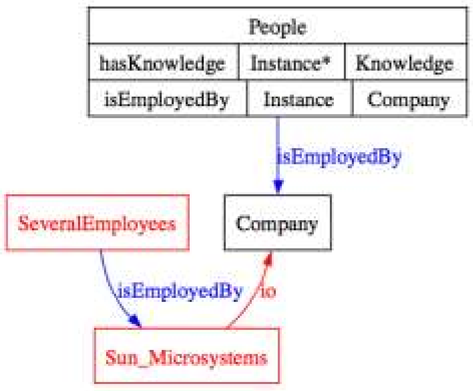


Figure 3.19: *Resulting picture from Ontoviz plug-in (with above options)*

In this image you can see that ***SeveralEmployees*** is linked to ***Sun_Microsystems***, which is a COMPANY, through the *isEmployedBy* property.

3.3.2 Choosing a Platform

The key to finding the answer to the question "Which systems is qualified best?" is in selecting the classes and individuals that should be displayed, and to set the options of the Ontoviz plugin in a way, that the desired relationship can be seen.

However, these images can become quite difficult to read if too much information is packed into one image. As an example for this the whole ontology has been depicted in image 3.20.

So the information that is desired has to be selected, for instance in image 3.21 only the main classes can be seen. This way the images are much clearer and the the relations between the elements can be seen and interpreted.

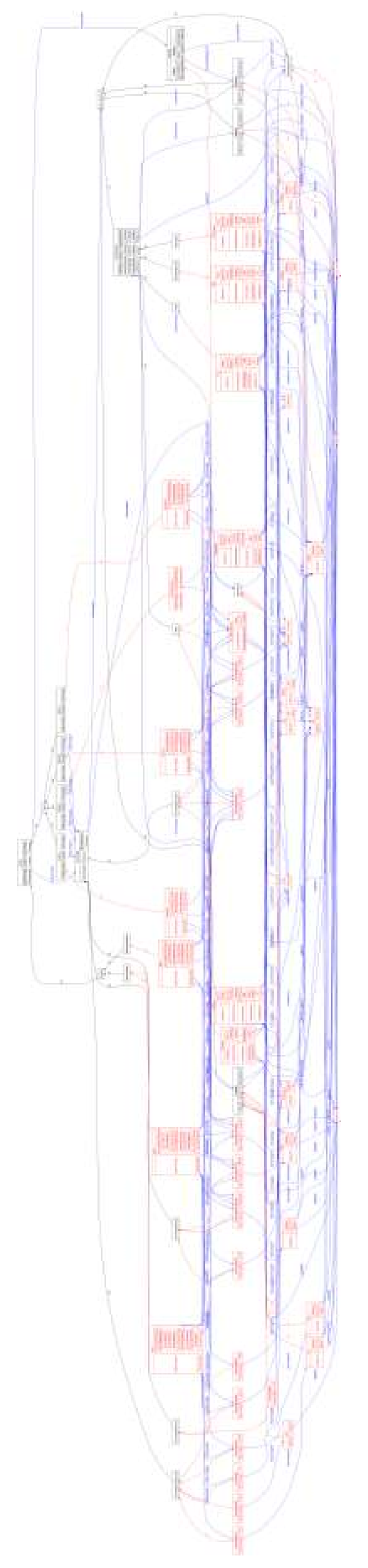


Figure 3.20: *An image of the whole ontology about the intranet infrastructure*

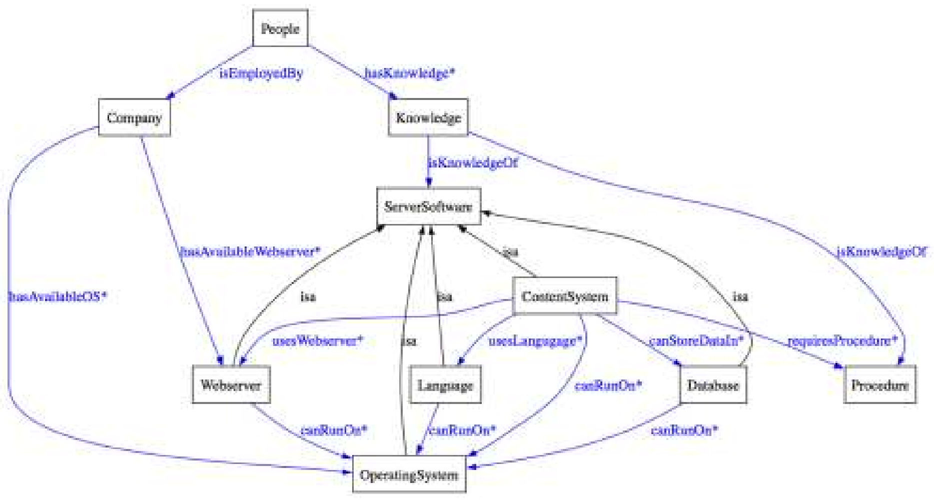


Figure 3.21: *Main classes of the ontology*

In image 3.22 the relationship between ***Sun_Microsystems*** and ***Typo3*** is displayed. From the image we see that ***Typo3*** can be operated on ***Apache*** which Sun Microsystems does provide. ***IIS*** which also supports Typo3 is left out, as Sun does not provide this. By this dynamic inclusion of Ontoviz only relevant information is displayed.

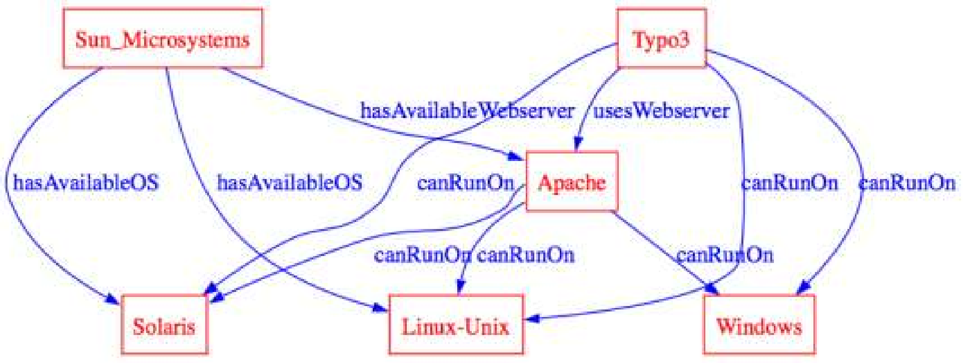


Figure 3.22: *Individuals **Sun_Microsystems** and **Typo3** and their relationships*

Now the connection from ***Sun_Microsystems*** and **WIKI** systems will be examined. Looking at image 3.23 it becomes obvious that ***FlexWiki*** has no connection to the technologies provided by Sun, while ***PmWiki*** and ***MediaWiki*** are linked to them. This means that ***Sun_Microsystems*** does not provide the necessary technology to run ***FlexWiki***, or the other way round ***FlexWiki*** does not meet the requirements of ***Sun_Microsystems***.

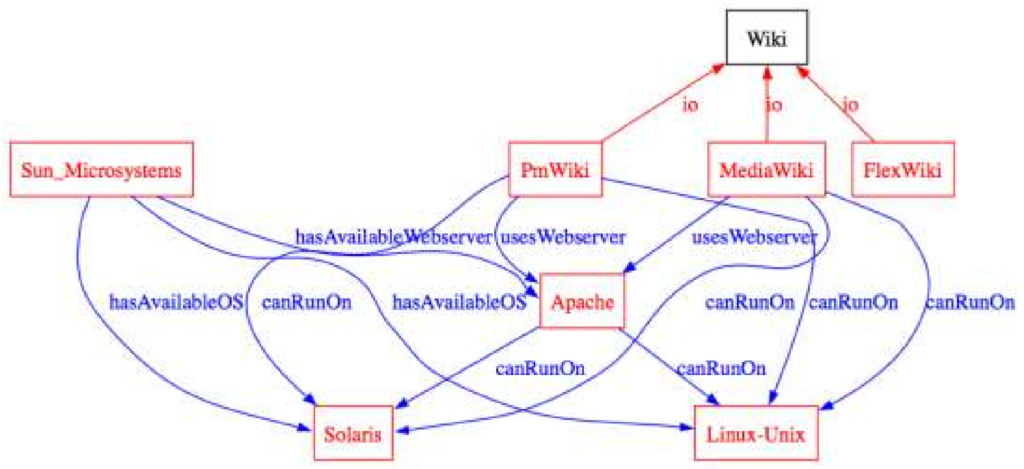


Figure 3.23: **WIKI** *system individuals and* ***Sun_Microsystems***

The actual decision was based on several such images and evaluations of them. In the exemplary case of ***Sun_Microsystems*** not only operating system and web server software was evaluated, but also the following:

- Skills necessary to operate platforms
- Knowledge of skills possessed by employees
- Procedures required to use platforms

Result Printing all schemes that were used to produce different views on the ontology would not provide additional insight in the methods used, but only bloat this paper. However the information was evaluated and Sun Microsystem settled on ***PmWiki*** as platform for its intranet.

Yet, it must be admitted, that the approach taken here turned out as suboptimal, because the images got quite complex and difficult to read, and so the selection process was tedious.

Chapter 4

CAP Ontology

Introduction This chapter details the second ontology, which tries to record all parameters of Sun Microsystem's "Change Acceptance Program" (short CAP). In the first section Sun Microsystems as a global company and the "Change Acceptance Program" itself are introduced. Afterwards the use of an ontology to capture the status of a CAP will be explained. This way an ontology could become an additional tool for the project manager who is conducting the CAP. In the third section, the details of the ontology will be displayed. And concluding, an evaluation of this tool will be discussed.

4.1 About Sun Microsystems and the CAP

4.1.1 About Sun Microsystems

Sun Microsystems provides network computing infrastructure solutions that include computer systems, software, storage, and services. Its core brands include the Java technology platform, the Solaris operating system, StorageTek and the UltraSPARC processor.

By investing in research and development Sun creates products and services that address the complex issues that customers face today, including increasing demands for network access, bandwidth and storage being driven by explosive growth in network participation and sharing. We innovate at all levels of the system and we partner with market leaders to provide value and choice for our customers.

Sun's network computing infrastructure solutions are used in a wide range of industries including technical/scientific, business, engineering, telecommunications, financial services, manufacturing, retail, government, life sciences, media and entertainment, transportation, energy/utilities and healthcare. (from [Micb])

4.1.2 Description of the CAP

CAP stands for "Change Acceptance Program". This program is an internal collection of processes and methods from Sun Microsystems to enhance the acceptance of new tools and workflows by its employees. A CAP is usually conducted as a project parallel to the project that introduces or produces the new tool and/or software, so that at the the common end the new tool can be introduced and, at that point in time, is already accepted by the future users of it.

This is usually done through a variety of different methods, which are better detailed in chapter 4.3 where the classes of the ontology are extended.

4.2 Motivation

In Quality Management acceptance of a project's outcome is at least as important as the quality of its results. The formula for this context is:

$$Q * A = E$$

Q = Quality, A = Acceptance, E = Efficiency

where Q is the quality of the projects result (e.g. the software produced), A is the acceptance of the presented solution, and E is the resulting effectiveness. Sun's CAP is supporting the effectiveness by helping the project manager to increase the acceptance. So in a CAP one will find methods and tools to enhance the acceptance rating of a project's outcome. A key conclusion is that a solution with a quality of 80 % that has an acceptance of 20 % is more worth than a solution with 100 % in quality with zero acceptance.

In these terms it is profitable to deduct some manpower from the actual project to do a CAP, as the outcome will be more effective. Doing a CAP is not an integral part of the original project, but present methods and tools which have to be executed simultaneously to the original project.

Objectives of CAP as printed in course materials [Mica] are:

- Understand the nature of organizational change and transition, so that tools and methodologies can be applied appropriately

- Understand and be able to describe the Sun CAP model, each of the individual elements, their intent and the linkages to each other.
- Be able to utilize Sun CAP in your work, enabling you to assess a change situation and apply appropriate elements of the model and tools.
- Be able to facilitate selected tools effectively in order to foster the right conversations and acceptance in a change setting

The methods of Sun's CAP are detailed in the educational material from Sun's internal courses [Mica]. These methods outline the activities that are required to enhance the acceptance of involved persons or groups.

Located in the center of the methods of CAP is *Mobilized Commitment* and arranged around it are the methods of CAP, which are:

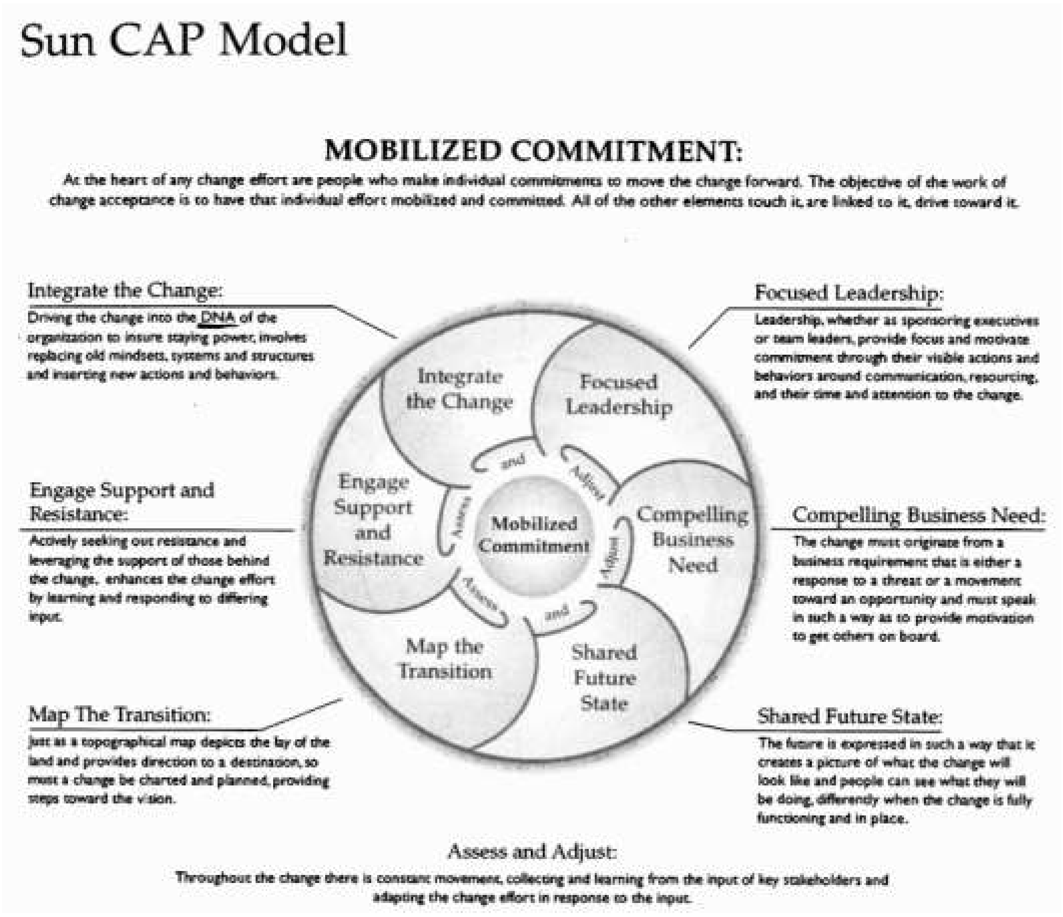
- Focused Leadership
- Compelling Business Need
- Shared Future State
- Map the Transition
- Engage Support and Resistance
- Integrate the Change

As these are the key elements of the ontology, they will be detailed in chapter 4.3. A view from the course materials can be seen in image 4.1.

4.3 Details of the Ontology

Again, we will take a closer look at the elements of the ontology. In the following sections the classes will be examined together with their properties and individuals.

Properties Before the individual classes and individuals are listed, properties should be viewed, as they have a slightly different function in this ontology, compared to the ontology from chapter 3. For the highly dynamic process of executing a "Change Acceptance Program" (CAP) actions have to be taken. Usually the result is obtaining values and results that can be interpreted.


Sun CAP Model

MOBILIZED COMMITMENT:

At the heart of any change effort are people who make individual commitments to move the change forward. The objective of the work of change acceptance is to have that individual effort mobilized and committed. All of the other elements touch it, are linked to it, drive toward it.

Integrate the Change:

Driving the change into the DNA of the organization to insure staying power, involves replacing old mindsets, systems and structures and inserting new actions and behaviors.

Focused Leadership:

Leadership, whether as sponsoring executives or team leaders, provide focus and motivate commitment through their visible actions and behaviors around communication, resourcing, and their time and attention to the change.

Engage Support and Resistance:

Actively seeking out resistance and leveraging the support of those behind the change, enhances the change effort by learning and responding to differing input.

Compelling Business Need:

The change must originate from a business requirement that is either a response to a threat or a movement toward an opportunity and must speak in such a way as to provide motivation to get others on board.

Map The Transition:

Just as a topographical map depicts the lay of the land and provides direction to a destination, so must a change be charted and planned, providing steps toward the vision.

Shared Future State:

The future is expressed in such a way that it creates a picture of what the change will look like and people can see what they will be doing, differently when the change is fully functioning and in place.

Assess and Adjust:

Throughout the change there is constant movement, collecting and learning from the input of key stakeholders and adapting the change effort in response to the input.

Figure 4.1: *Overview of Sun's CAP methods as displayed in course materials [Mica]*

To be able to describe the need to carry out a special action, properties are used in the ontology. For example in the **MOBILIZEDCOMMITMENT** class the property *identify_Stakeholders* signals the need to identify the stakeholders from the persons involved in a project. Sometimes the property is linked to the class that is obtained by executing the action behind the property. In this case this is the **STAKEHOLDERS** class. See this context in image 4.2.

4.3.1 MOBILIZEDCOMMITMENT class

The **MOBILIZEDCOMMITMENT** class represents the central goal the project manager has, when conducting Sun Microsystems' CAP. It is merely in the ontology to allow other, more functional elements, to be arranged around this "central" class and to be connected to it. Therefore it has one property only: *identify_Stakeholders*. Image 4.2 displays this class.

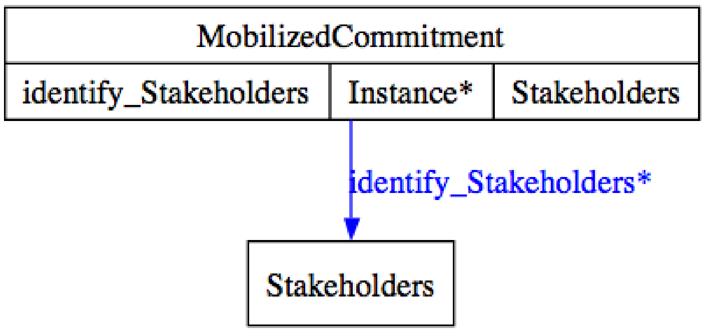


Figure 4.2: **MOBILIZEDCOMMITMENT** *class*

4.3.2 PROJECT class

The **PROJECT** class is used to administer different projects along with their parameters. In image 4.3 this class can be seen along with an individual, that could represent the project "Intranet Migration" from section 3.

4.3.3 CAP-SECTIONS class

In image 4.4 the first level of subclasses from **CAP-SECTIONS** class can be seen. They represent the main methods from CAP (compare image 4.1). In detail (as mentioned before) these are:

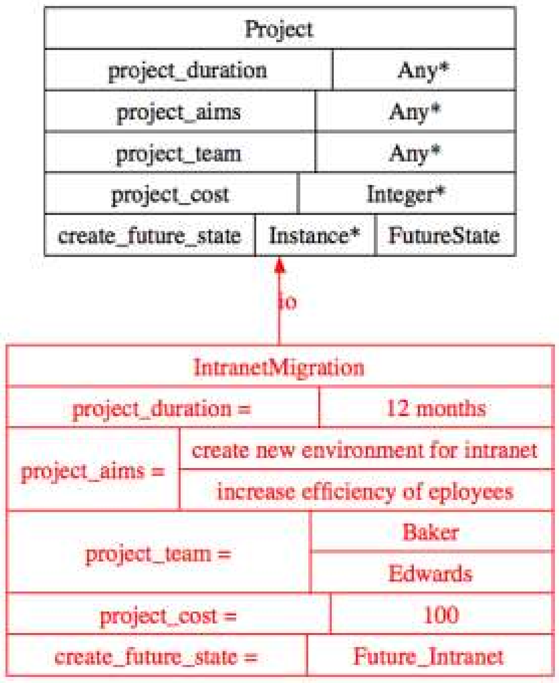


Figure 4.3: **PROJECT** *class*

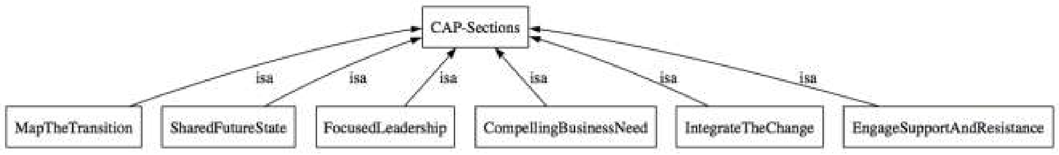


Figure 4.4: *Main Sections of CAP as classes*

FOCUSEDLEADERSHIP To perform any sort of change, leadership is necessary. The concepts of this class deal with two sorts of leadership: external and internal. External leadership concepts deal with persons, that are leaders, either by position (superiors) or behavioral (motivators). External leaders have to be convinced, as they have substantial influence on others, and therefore have an impact on the transition. Internal leadership refers to leading the project team, but also to enable the project team members to become leaders themselves, by motivating others. The class is depicted in image 4.5.

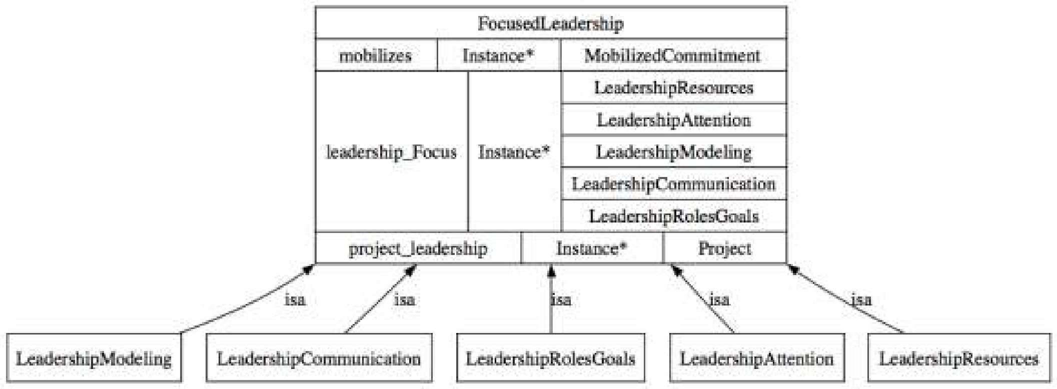


Figure 4.5: **FOCUSEDLEADERSHIP** *and its concepts*

The following subclasses of this class stand for concepts that are useful to control the work that has been accomplished in convincing leaders of the change. They are part of both kinds (internal and external) of leadership unless stated otherwise.

LEADERSHIPRESOURCE Is a concept of external leadership only, and deals with materials, money and people. In individuals of this class the parameters of individual leaders can be stored.

LEADERSHIPMODELING The way of interacting with others in terms of the project is called **LEADERSHIPMODELING**. The individuals of this subclass can store changes in the attitude towards the project.

LEADERSHIPCOMMUNICATION The concept of leadership communication is the art of promoting the project and to give support to it publicly (external leadership) but also refers to the approaches of intra-team communication, de-escalation of problems and reporting (internal leadership).

LEADERSHIPATTENTION To enhance the publicity the **LEADERSHIPATTENTION** class is used. This is basically the same in external and internal leadership. This class deals with meetings for instance, and the time reserved for the project in them.

LEADERSHIPROLESGOALS A concept present only in internal leadership is **LEADERSHIPROLESGOALS**. Here milestones and roles of team members are defined.

COMPELLINGBUSINESSNEED This class (seen in image 4.6) symbolizes the business case that can be created to convice the stakeholders of the necessity of the change. Basically there are two subclasses that need to be identified for each project. With them the manager of the CAP has the reasoning provided to convince decision makers to adopt the proposed change. Those two subconcepts, that can be instantiated to provide precise parameters of a project, are **THREATSIFNOCHANGE** and **OPPORTUNITIESIFCHANGE**, and basically they stand for the advantages that are gained by adopting the change, and the disadvantages that will be faced if the changed will not be conducted.

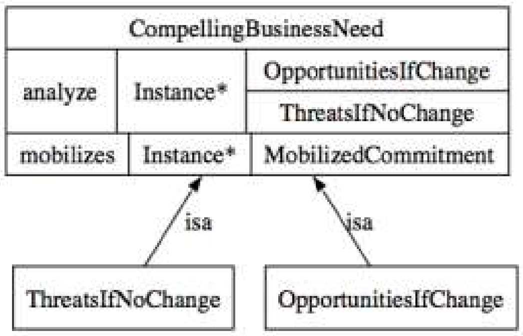


Figure 4.6: **COMPELLINGBUSINESSNEED** *and its concepts*

SHAREDFUTURESTATE Visualizing a future state does help in creating direction and focus of the transition. It helps in directing the energy. To optimize the energy spent by all involved persons, the future state that is imagined by them should be the same, they should have a "shared future state". An important part of the future state are the advantages gained by the new methods through the project. They have been gathered in the "compelling business need" part of the CAP. The associated class along with the **PROJECT** class and their corresponding individuals can be seen in image 4.7.

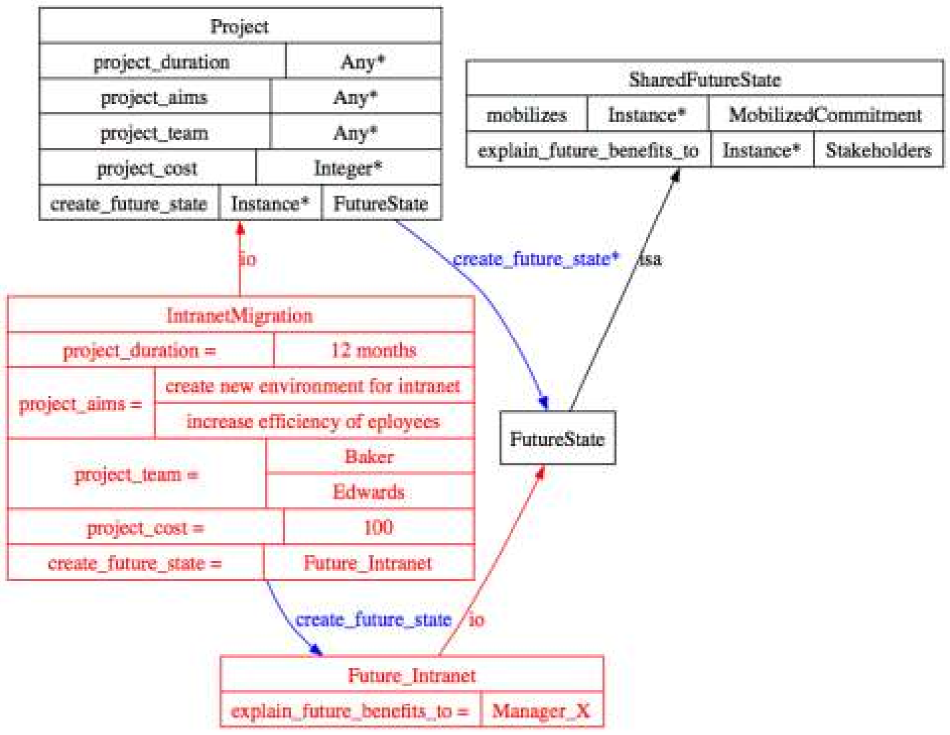


Figure 4.7: **SHAREDFUTURESTATE** *and* **PROJECT** *with individuals*

MAPTHETRANSITION Now, as the future state is clear a path from the current state to this specific future has to be mapped. This is attained by the "map the transition" method. It focuses on the major elements of the change, and helps to avoid going astray. The map also serves as guide to place certain elements of the transition, and an attached timeline can show the progress and motivate to contribute to the project.

Current state To be able to draw the path from "now" to "then" the current state has to be analyzed and formalized. Two concepts are significantly involved in the current state: **CONTEXT** and **CULTURE**.

CONTEXT "Context" relates to current conditions in the department or company that is about to experience the change. These are the parameters that exist in general and for every project (i.e. resources, personnel, competitors, strategy).

CULTURE In contrast to "Context", "Culture" relates to the internal conditions in a group of people. It is the way how communication is done and how a

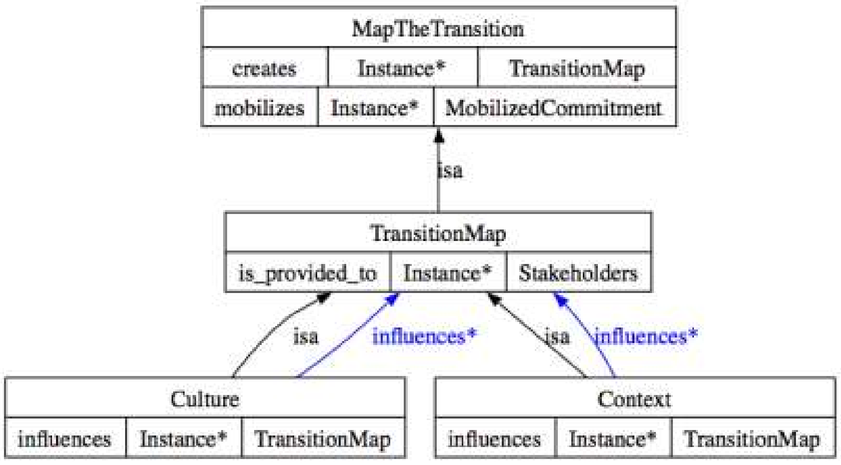


Figure 4.8: MAPTHETRANSITION *with* TRANSITIONMAP

group of peaople is organized. In short it is "the way, how people do things" in a company or department. In image 4.8 these concepts can be seen.

ENGAGESUPPORTANDRESISTANCE After the path to the future state is defined, those who support or resist the change have to be identified and dealt with. Especially convincing the resisters is very important to the success of the project. But also helping the supporters to improve their performance is well invested energy. In image 4.9 the class ENGAGESUPPORTANDRESISTANCE is displayed.

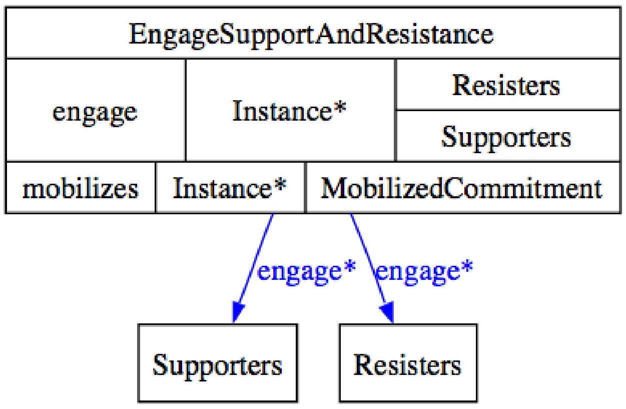


Figure 4.9: ENGAGESUPPORTANDRESISTANCE *class with subclasses*

INTEGRATETHECHANGE This last method of CAP ensures that the change is adopted to the structures and organisation of the company, and thus has the power to stay. It cares about the peoject after it has been officially terminated. There has to be someone who is accountable, even after the project has ended. The software or workflow (or whatever the outcome of the project was) may need amendments or maintenance or the users may need support using it. Maybe the skills to maintain the results are not in the company yet, so someone has to be hired for this task.

4.3.4 PERSONS class

People are important to every project. They appear as members of the team, stakeholders, users, supporters, manager, deciders etc. This class is meant to hold the individuals that are encountered throughout the project and, seeing how they fit in, organize them in subclasses. The **PERSONS** class can be seen in image 4.10.

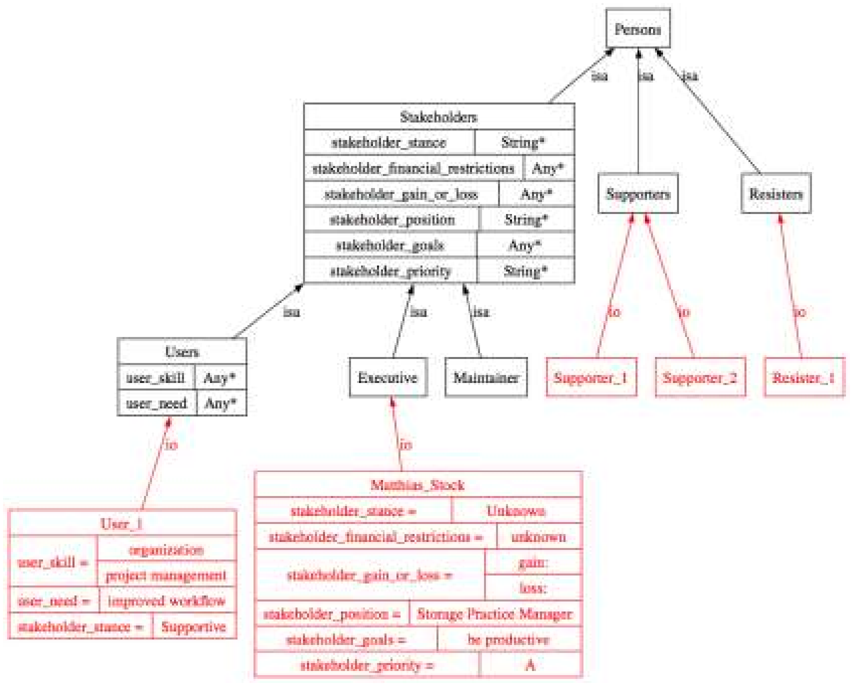

Persons
isa
isa
isa
Stakeholders
stakeholder_stance String*
stakeholder_financial_restrictions Any*
stakeholder_gain_or_loss Any*
stakeholder_position String*
stakeholder_goals Any*
stakeholder_priority String*
Supporters
Resisters
isa
isa
isa
io
io
io
Users
user_skill Any*
user_need Any*
Executive
Maintainer
Supporter_1
Supporter_2
Resister_1
io
io
User_1
user_skill = organization project management
user_need = improved workflow
stakeholder_stance = Supportive
Matthias_Stock
stakeholder_stance = Unknown
stakeholder_financial_restrictions = unknown
stakeholder_gain_or_loss = gain: loss:
stakeholder_position = Storage Practice Manager
stakeholder_goals = be productive
stakeholder_priority = A

Figure 4.10: **PERSONS** *with several exemplary subclasses and individuals*

4.4 Results & Evaluation

4.4.1 Introduction

Change is an important part of our life. Only the individual that is able to change, to adopt to new surroundings will prevail. Regrettably too few project teams deal with the acceptance of their products or results. It does not depend on whether the actual result of a project is a piece of software, a new kind of workflow or a new partnership with another company. If this change is not accepted by the persons and teams that have to work with this new, changed environments failure is predetermined.

Sun Microsystems' CAP is a collection of tools that may appear time-consuming, but the invested time is invested profitably. Conducting a CAP is a project parallel to the actual project, and does consume time and resources, but the formula from the beginning of this chapter,

$$Q * A = E$$

Q = Quality, A = Acceptance, E = Efficiency

makes it obvious that a solution can have any degree of quality, but if the acceptance is close to zero, the efficiency approaches zero as well. So the acceptance of a solution can not be overestimated, and at least the major elements of the CAP should find its place in every project schedule. Now the individual elements will be reviewed and summarized.

MOBILIZEDCOMMITMENT and its connections The central task of a "Change Acceptance Program" is to enhance user acceptance, to mobilize commitment from the users. So everything that is done, focuses on **MOBILIZEDCOMMITMENT**. But how are the other concepts and methods connected to the "Mobilized Commitment"? How do they accomplish mobilizing commitment? The following paragraphs answer these question, and he classes relationship can be seen in image 4.11.

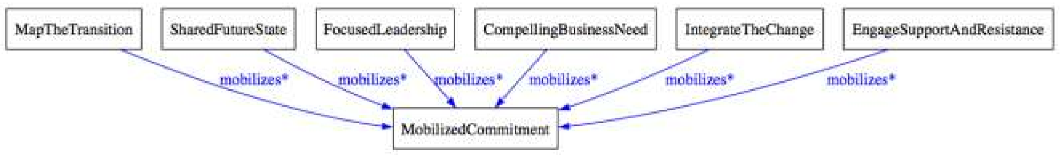


Figure 4.11: **MOBILIZEDCOMMITMENT** *and its connections*

FOCUSEDLEADERSHIP To help the employees of a department or company understand the benefits of the changes that are introduced by the project focused leadership is necessary. The demonstration of commitment by leaders will increase the enthusiasm of co-workers.

COMPELLINGBUSINESSNEED To compel the support of involved individuals the business need is displayed in a radical way. The according pros and cons should suffice to gain the support of key leaders, but also to create excitement among the project team.

SHAREDFUTURESTATE The tension between the current and the future state drives the commitment. Inspiring the users and the project team who can see the impact on their specific surrounding, the future state does not only state the goals of the current project, but also relates them to the long term goals and philosophy of the department or company. To be constantly aware of the future state is very important for the success of the project.

MAPTHETRANSITION Keeping focused on the path to the future state is important. The transition map created in this method helps to accomplish this. In addition to that, having the path laid out helps to keep the spirit up, as milestones can be tracked easily.

ENGAGESUPPORTANDRESISTANCE Resisters can hinder the progress a lot. So possible resisters have to be engaged early. Sometimes they turn into powerful supporters and can help with mobilizing others' commitment. Also existing supporters can be helped in finding the best way to support the change, because without guidance some people that are willing to help do nothing because they don't have the confidence or knowledge to do so.

INTEGRATETHECHANGE Finally this section is the closest to the actual project. It interacts with it on several levels. The methods presented in this part check if important topics were covered, like people understand the change and it does not surprise them. All individuals received the training they needed and new employees were hired as necessary to provide missing skills.

4.4.2 Using an Ontology for a CAP

This thesis examined the possibility to use an ontology to keep track of an ongoing CAP. The benefits of this are having the current status available and being able to share it. The interesting parts to keep the current state are:

- Stakeholders and their stance

- Goals of stakeholders and users
- Achieved project results

Beyond that, the project leader should be able to see the necessary actions from the ontology based on the data above. How would that look like? In image 4.12 an example of an stakeholer ***Matthias_Stock*** is given. You can see the ***Intranet-Migration*** project and the **FUTURESTATE** ***Future_Intranet*** which are related to ***Matthias_Stock***. A change to the parameters (for instance the stakeholders stance towards the project) can be kept up to date by the objects data properties, in this example *stakeholder_stance* can be changed from “unknown” to “supportive” or “against” depending on his attitude.

For each project these individuals have to be created, as each project will have different parameters like stakeholders and project team.

It was discovered during the course of this thesis that looking up and controlling the status of individual elements was no easy task, as either the corresponding images had to be created first, and were difficult to read sometimes, due to the large amount of information. If the use of images was abandoned the ontology bestowed no additional value to the current methods of spreadsheet and tabular storage of status information.

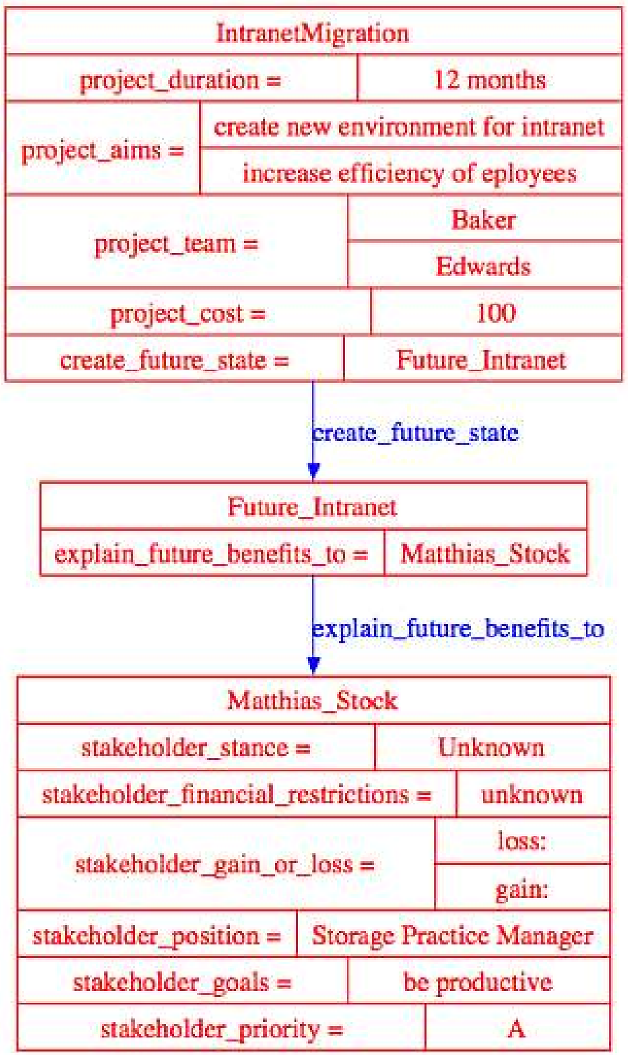


***Figure 4.12: Matthias_Stock** with properties*

Chapter 5

Conclusions

"Change is the essential process of all existence."

Commander Spock, Episode: "Let That Be Your Last Battlefield" [MA]

Introduction Without changes mankind would not exist as we know it today. Yet even evolution itself does try different paths (at least in regard to biology) before the best branch is developed further. Earth and the whole universe have always worked this way. In context of information science evolution is not that self-evident. Humans have to develop new ideas and techniques actively and, if several options exist, decide on the one to use.

5.1 Summary of Results

5.1.1 Intranet Ontology

In this problem a way to decide on a new infrastructure of an intranet was searched. With so many different software solutions at hand, which one is the best? Which one fits the needs best?

This thesis tried to use an ontology to record all parameters that are influential to the decision and then make this decision with the help of the ontology.

While the ontology provided an excellent way of visualizing details of the ontology (see image 5.1), the display of the whole system of classes and individuals was possible but did not make sense (see image 5.2). Therefore the inspection of single correlations was easily possible and resulted in usable information. Yet the effort needed to split the original task in many small queries and combine the answers together again, canceled the benefits gained in the first place.

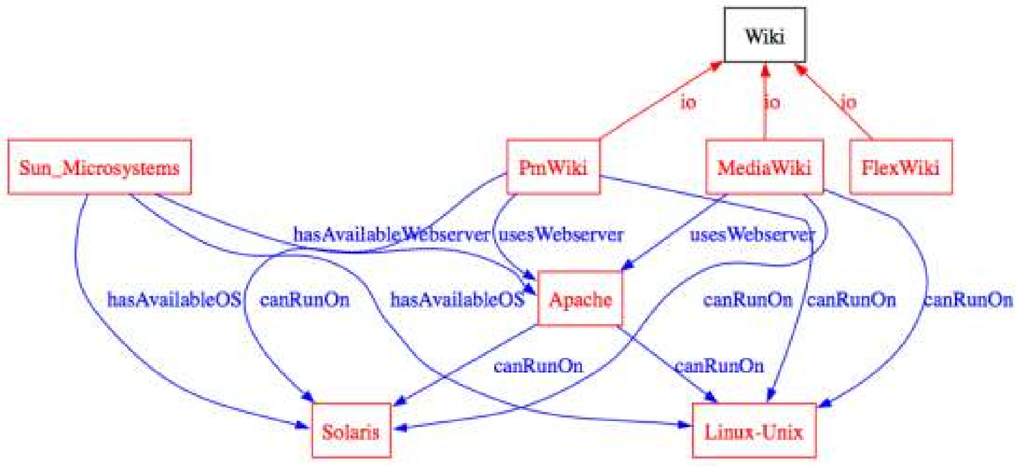


Figure 5.1: *WIKI system individuals and* ***Sun_Microsystems***

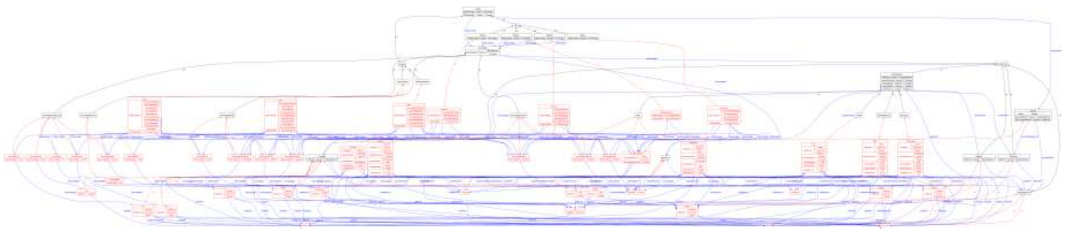


Figure 5.2: *An image of the whole ontology about the intranet infrastructure*

However, creating the ontology helped in identifying the necessary parameters to decide on the right platform, yet the original idea of visualizing the whole system of company, employees, server and software had to be abandoned, because the resulting pictures were not legible.

5.1.2 CAP Ontology

The second ontology dealt with Sun Microsystems' "Change Acceptance Program" (CAP). This program provides methods and approaches to increase the acceptance of a change. These methods usually are applied at the same time as the project takes place, that implements the change. Amongst other principles the CAP convinces users and other stakeholders of the necessity and benefits of the change.

In the CAP are a lot of processes that can be executed. To track those and to track the attitude of persons towards the project was the goal of this ontology. Effectively one of the problems that was encountered is similar to the main problem of the ontology about the intranet infrastructure. The user of the ontology has to cope with the legibility of the graphics. The tradeoff between information content and legibility leads to a unsatisfying situation. Either only few information can be covered or the interpretation is too difficult.

A full "Change Acceptance Program" is seldom performed. Mostly the project leader picks elements from the offered methods to address specific needs. So if he used the ontology totrack the status, he would carry along a lot of unused information. An another note, if the benefits that are gained from an ontology can only be accessed if additional software is installed (an ontology editor) and the use of the new software has to be learned prior to using ontologies as tools, then even less people will use CAP methods at all.

5.2 Concluding Remark

Can ontologies be brought to effective use at all? While this might seem a legit question, the answer will certainly depend on the usage. According to the results of this thesis however, ontologies can not be used for every data-driven decision that has to be made. The Fraunhofer Institute for Software and Systems Engineering (ISST) wrote in its annual report for the year 2003 that a lot of "modern" software and system solutions are labeled with "Ontology" or "ontology-based" only to indicate that the product has a reasonable concept ([Ges03]), but they seldom make actually use of ontologies.

While for some usages it does make sense to use ontologies, you should not lose track of the original objective of ontologies: data sharing and unification of information for re-use ([UG96]).

Appendices

List of Figures

Bibliography

[Cos] CosmoCode. Wiki matrix: Wiki engine comparison tool. http://www.wikimatrix.org/. 3.1

[Cun] Ward Cunningham. Correspondence on the etymology of wiki. http://c2.com/doc/etymology.html. 1.1

[Ges03] Fraunhofer Gesellschaft. Fraunhofer gesellschaft – institut software- und systemtechnik – jahresbericht 2003. http://www.iuk.fraunhofer.de/downloads/jahresberichte/ISST_Jahresbericht_2003.pdf, 2003. 5.2

[Gru] T. Gruber. What is an Ontology. http://www-ksl.stanford.edu/kst/what-is-an-ontology.html. 2.1

[JB] CEO Netscape 1995 1999 Jim Barksdale. Jim barksdale home page at netscape.com. http://people.netscape.com/jimb/index_40.html. 1

[Lor] J. Lorhard. Ogdoas scholastica. 2.1

[MA] the Star Trek Wiki Memory Alpha. Let that be your last battlefield. http://memory-alpha.org/en/wiki/Let_That_Be_Your_Last_Battlefield. 5

[Mat] CMS Matrix. The content management comparison tool. http://cmsmatrix.org/. 1.1, 3.1

[MH$^+$04] Alan Rector Matthew Horridge, Holger Knublauch et al. A Practical Guide To Building OWL Ontologies Using The *Protégé* -OWL Plugin and CO-ODE Tools, Edition 1.0. http://www.co-ode.org/resources/tutorials/ProtegeOWLTutorial.pdf, 2004. 2.2, 2.2, 2.3, 2.3, 5.2

[Mica] Sun Microsystems. Sun CAP Application Session - Participant Guide. Internal Educational Material. 4.2, 4.1, 5.2

[Micb] Sun Microsystems. Sun microsystems - about. http://www.sun.com/aboutsun/company/index.jsp. 4.1.1

[MW] Merriam-Webster. OnLine Dictonary, Defintion of Ontology. http://www.m-w.com/dictionary/ontology. 2.1, 2.1, 5.2

[Sun] Sun Microsystems GmbH Intranet. http://sunweb.germany, intranet access only. 3.1, 3.1, 3.2, 3.3, 5.2

[UG96] M. Uschold and M. Gruninger. Ontologies: Principles, Methods and Applications. *To appear in Knowledge Engineering Review*, 11(2), 1996. 5.2

[Unia] Stanford University. Ontoviz visualization plug-in for *Protégé* . http://protege.cim3.net/cgi-bin/wiki.pl?OntoViz. 2.4, 3.3.1

[Unib] Stanford University. *Protégé* : free, open source ontology editor and knowledge-base framework. http://protege.stanford.edu/. 2, 2.4

[Vii] M. Viinikkala. Ontology in Information Systems. http://www.cs.tut.fi/~kk/webstuff/Ontology.pdf. 2.1

Content of attached CD

DIRECTORY / FILE(S)	CONTENT
`/README`	Contents of the CD and readme file
`/diploma_thesis_lederer.pdf`	Diploma thesis in PDF format
`/progress_report_lederer.pdf`	Progress report, March 20th 2007
`/latex/`	Diploma thesis in LaTeX format
`diploma_thesis_lederer.tex`	Main LaTeX document
`diploma_thesis_lederer.bib`	Bibliography file
`figures/`	Included image files
`section/`	LaTeX sub-parts of thesis (chapters)
`structure/`	LaTeX sub-parts of thesis (frame)
`/ontologies/`	Ontology files created for this thesis
`intranet.owl`	Intranet (Sec. 3) – ontology
`intranet.pprj`	Intranet (Sec. 3) – *Protégé* file
`cap.owl`	CAP (Sec. 4) – ontology
`cap.pprj`	CAP (Sec. 4) – *Protégé* file