Teaching Global Software Engineering
A Practical Experience

Florian Matthes, Christian Neubert, Christopher Schulz (Eds.)

Software Engineering for Business Information Systems (sebis)
Ernst Denert-Stiftungslehrstuhl
Chair for Informatics 19
Technische Universität München

Boltzmannstraße 3, 85748 Garching b. München, Germany

wwwmatthes.in.tum.de

Abstract

Since software is no longer developed by one enterprise which is located at one single site only, modern software engineers have to strive for distinct skills and capabilities allowing them to work together on a global scale. When conjointly designing, implementing, and testing different software components, these distributed engineers will also have to contribute their local know-how and country-specific experience in order to guarantee the final success of the software project. As a prominent branch of research regarding this world-wide software engineering process, the discipline of Global software engineering (GSE) has gained increased attention over the past years. However, when taking a closer look on today's universities curricula, the transfer and intermediation of GSE can be presently seen as an emerging activity.

This report presents the main results and lessons learned when teaching GSE by means of the execution of four software engineering projects at Technische Universität München during the winter term 2009/10. The overall objective of these projects was to allow globally distributed student teams to work conjointly on a software engineering task aiming at enhancing their project management and communication skills by taking differences in geographical location, academic curriculum, and culture into consideration. In depicting the general outcome of the four projects in the form of the individual student report, the document describes both, the specific content of each GSE project as well as the organizational issues student members were facing in the course of their work. Additionally, the report provides guidance for future GSE activities at universities by pointing out the experience made and the knowledge gained on the part of the teaching staff.

Contents

CHAPTER 1

Introduction

1.1 General motivation

Many technological, organizational, and economic factors have led to the increased globalization of development projects [SMP06]. Important business drivers for *Global Software Engineering* (GSE) include cost competitiveness, access to talent regardless of location, the need for a globalized presence, as well as mergers and acquisitions [Ca99, FMS10]. Globally-distributed projects are rapidly becoming the norm for large software systems [He07]. GSE imposes new challenges on software engineers, in which coordination over distance can be considered as one of the major ones [He07, LH09, LHB07]. In this respect, there is a strong aspiration for educating and training IT employees to act conjointly in an international environment. Not only those engineers are geographical separated from each other by developing at multiple sites on often tightly interconnected modules, they are also confronted with cultural differences, e.g. national culture, native language, as well as organizational culture [Ca99, HH04]. Whatever the spur for GSE, pure technical know-how and engineering craftsmanship in the area of development are essential but the ability to collaborate, both efficiently and effectively, in globally distributed teams is equally important [Go09].

When taking a closer look on the personal requirements modern software engineers are challenging in their daily work life, it is astonishing to observe that teaching Global Software Engineering at academic institutions is still in its infancy [LMM08]. Besides its recent occurrence and emerging significance in an industry context, the sparse offering of intermediate GSE at universities maybe caused by the additional coordination overhead hosting institutions have to cope with when organizing such type of practical courses. Furthermore, unsynchronized semester schedules in addition to different grading and evaluation schemes significantly impede the cooperation among global distributed universities.

In the light of aforementioned situation, Prof. Laurini from INSA de Lyon, France initiated

a university course coined by the name *Network of Engineering univeRsities Educating in Intercultural Design* (NEREID) [NP10] in July 2008. The cooperative course targeted at teaching students how to work on software engineering projects including the geographical dimension of separated teams. Therein, this document presents four final reports elaborated by globally distributed student teams of 4-6 persons when working on one software engineering task of the NEREID course. More precisely, the report provides teaching staff with the final deliverable of four concrete examples of global software engineering projects, which have been successfully carried out by late bachelor and master students in winter term 2009/10 during a three months long time frame.

1.2 Historical background

Initially, the NEREID course was launched by Prof. Robert Laurini in the Information Technology department of INSA de Lyon in France. Considering the typical curriculum of the master computer science graduates of INSA, six students work conjointly on six projects during a given time span of one year. In this vein, each student has to assume the role of the project lead, following the thought that an expert in computer science must not only be a good programmer, but furthermore has to obtain distinctive skills in project management, especially for the design and coding of huge software products. However, the main disadvantage is that local students possess the same culture, apply similar methodologies, and speak identical languages. Regarding the future work of those students, they will much likely be in contact with colleagues having different cultures, speaking different languages, as well as mastering different methodologies.

Since INSA regards international project experience as a focal point of each student's curriculum [1], in fall 2007 the idea arose to organize cross-country projects, in which students from different countries would have to work conjointly on one software engineering project. Thereupon, Prof. Laurini and his colleagues contacted three European universities in 2008. Although, the institutions considered student exchange as being of the utmost importance, they all denied the proposal for several reasons. As for the reasons stated, either project management represented not their priority, time period and duration were not relevant, or project management was a priority, but organization could not comply.

In July 2008, Prof. Laurini was invited to the Tecnológico de Monterrey campus Puebla in Mexico, were the colleagues of the Mexican chair were immediately enthusiastic about his project ideas. As a fruitful consequence, one student project topic was selected, and two groups of France-Mexican students were organized in order to work together from autumn to winter 2008. In detail, the subject consisted in the creation of a small information system for a tourism office based on Google mash-ups. As a follow-up of this experimentation, a first post-mortem report was compiled [LS09], analyzing the different aspects encountered during the project execution in an international context. In addition, the two partners jointly decided to extend the NEREID course in terms of academic partners and number of student projects facing the upcoming year 2009.

[1] INSA de Lyon is a very international-oriented institution with more than 25% of its students being from foreign countries. Moreover, 75% of INSA students are spending one or two semesters abroad.

1.3 Participants and projects

To increase the scope of NEREID with regards to the participating partners and total number of student projects, Prof. Laurini contacted several colleagues at different academic institutions in winter of 2008. Finally, five chairs of five different universities speaking three different languages conjointly organized and executed a practical course in the winter semester 2009/10. The participating universities can be seen in fig. 1.1

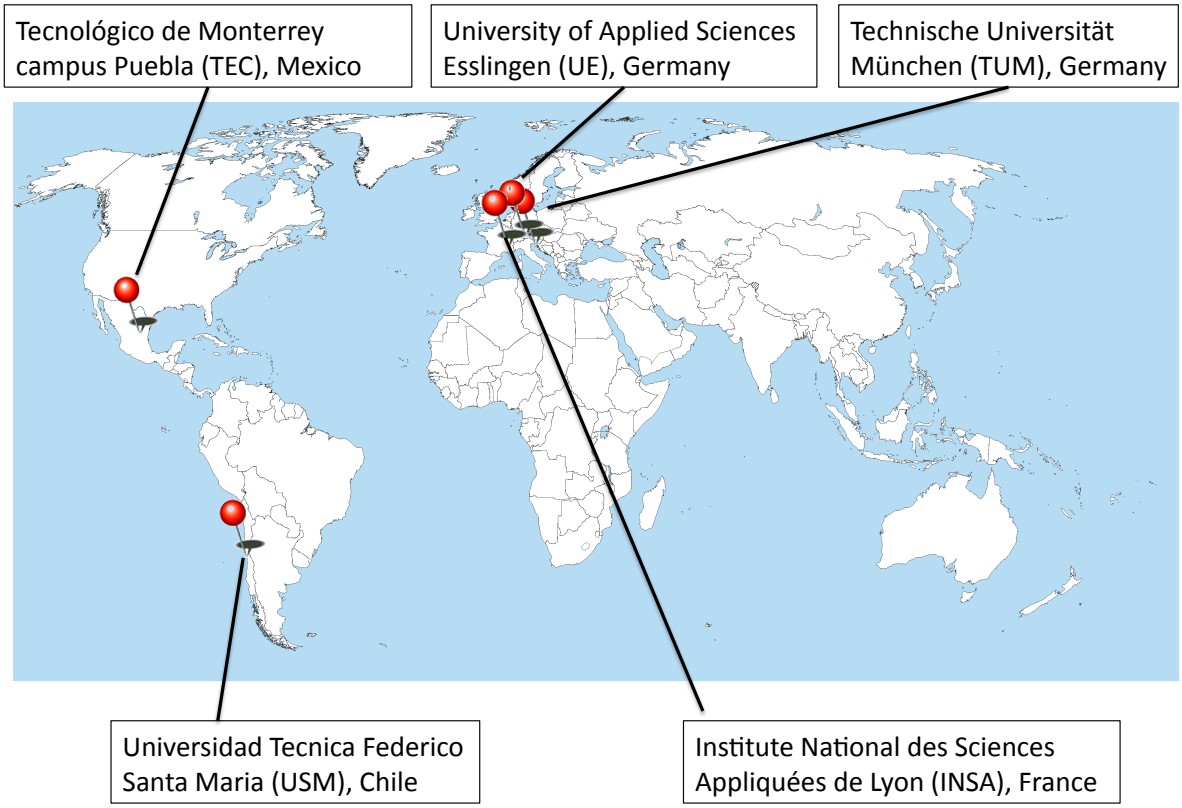

Figure 1.1: Participating universities

After the university members were identified, the student project tasks were worked out by the respective professors and their assistants. As for this document, the four different projects executed by Technische Universität München in the role of project host as well as tutor are presented in the following chapters, namely

- Picture-based itineraries (INSA),
- Cadaster system (TEC),
- Internationalization & localization of Tricia (TUM), and
- University relations maps (UE).

Note, that the hosting university is stated in the brackets. By depicting the final outcome of four student projects, this document therefore complements the publication by concrete deliverables being useful for further GSE activities.

1.4 Report structure

In depicting the final project reports of four different student teams in following chapters, this document helps in two ways: on the one hand, explicit suggestions for the content of global student projects are made which can be covered by a team of 4-6 students during a time frame of approximately three months. On the other hand, the lessons learned described in this document should help teaching staff to easily identify and overcome pitfalls we observed during the execution of the GSE projects on the part of the advisors as well as the students. Thereby, each of the four following project chapters is subdivided into an Introduction, a Project organization, a System Specification and development, as well as a Lessons learned section to facilitate reading and allow comparison among the different projects. Lastly, the Summary and Outlook chapter 7 concludes the report in highlighting the main lessons learned made by the teachers.

CHAPTER 2

Picture based itineraries

Author:	Florian Balke, TUM Markus Bauer, TUM
Project sponsor:	Prof. Robert Laurini, France
Participating universities:	Universidad Tecnica Federico Santa Maria (USM), Chile Institute National des Sciences Appliquées de Lyon (INSA), France Technische Universität München (TUM), Germany
Team members:	Sebastian Acevedo, USM Franz Fahrenkrog, USM Jean-Andrei Diaconu, INSA Sannine Saghbiny, INSA Markus Bauer, TUM Florian Balke, TUM

2.1 Introduction

The main task of the "Software engineering in international teams" seminar was to realize a software project in cooperation with students from other international universities. In our case of the "picture based itineraries" project, it was a cooperation of students from the INSA Lyon in France, the USM in Valparaíso in Chile and the TUM in Germany. The project was offered by Prof. Laurini from INSA de Lyon. As our professor in charge, he specified the project task and suggested the French team for project leadership. In the context of our project, Prof. Laurini could be regarded as the customer of the project.

The project task itself consisted of the development of a navigation system for pedestrians, in particular one which explains an itinerary by a sequence of pictures. This sequence of pictures was supposed to be assembled depending on the processing of a shortest path algorithm which determines the shortest route between the actual position of the pedestrian and the desired destination. To explain the route by the pictures, they had to be decorated with arrows pointing in the direction, the pedestrian has to walk. The client device had to continuously validate the position of the pedestrian using GPS or

related technologies which enables the navigation process. The arrows had to be calculated from the angle between the current and the following road, to achieve a convenient result. In addition the arrows were supposed to be silhouetted against the background, wherefore a contrasting color had to be determined.

2.2 Project organization

2.2.1 Project teams and responsibilities

Paying tribute to the closeness of the customer Prof. Laurinin ([Pr09]), we decided to give the role of the project leadership to the French team. The distribution of project subtasks happened during the first three meetings in which we also detailed these tasks and made the most important design decisions for the implementation and the used hardware. Unfortunately most of these decisions were made without participation of Chilean project members. Due to the general lack of time, it was not possible to wait on the first response of them which had implied a delay of the whole project.

The only common deadline was to finish the development before Christmas holidays which meant to get familiar with the used technologies, to implement each software part, to integrate all parts, and to test them within six weeks. According to this tough timetable we regarded to begin the integration phase latest two weeks before the deadline as a necessary and also achievable milestone. During the time before the integration, each team was responsible of itself for the development of their part. The weekly meetings took place for the progress reporting and possible adjustments of the specification.

2.2.1.1 French team: Leading Team

School and department:
INSA, Department of Informatics

Teachers:
Prof. Dr. Robert Laurini and Dr. Beatrice Rumpler

Responsibilities:
The leading team's role was to manage the coordination of the other teams, monitor the project, its initial planning (task assignment, costing time achieving each task) and make the necessary decisions in unforeseen circumstances involving the initial planning. Furthermore the French team was responsible for the server and the database. So their task was the deployment of a web server which calculates the itinerary in order to provide the necessary functionality for the client and to create an appropriate database schema.

2.2.1.2 German team

School and department:
TUM, Department of Informatics

Teachers:
Christopher Schulz and Christian Neubert

Responsibilities:
The German team was responsible for the client software of the project. The client had to be a handheld device which is able to determine the position using GPS and to access the Internet from

everywhere. These technologies are necessary for the navigation functionality because the business logic is offered through web services by the server and the navigation takes primarily place outdoor.

2.2.1.3 Chilean Team

School and department:
UTFSM, Department of Informatics

Teachers:
Prof. Dr. Jose Contreras

Responsibilities:
The semester of the Chilean team has already ended at the beginning of December. So they had much less time to work on the project than the other team members. Considering this fact, we all agreed to let them work on a smaller part of the project which can not provoke a total failure of the project. The Chilean team was responsible for the second bigger server part, the picture processing which decorates the pictures with arrows to explain where to go.

2.2.2 Project organisation and collaboration

The introduction meeting of the TUM for the seminar was on October 9th in which the German students could apply for one for the different projects. We decided to participate in the "Picture based itineraries project". Thenceforward, the first task was to get in touch with the customer of the project on the one hand and the other project members on the other hand.

The first contact with the other team members on October 22nd is regarded as the actual project start. Since that time onwards, the project work went through four general phases. At the beginning, the requirement elicitation that resulted in the system specification followed by the familiarization with planned technologies had to be fulfilled. After this preparation, the project went on to the implementation of the software which needed a link back to the specification because of a continuously deepening understanding of the used technologies. The finalization of the still separated software parts led to their integration and the testing of the integrated system.

2.2.2.1 Project phases

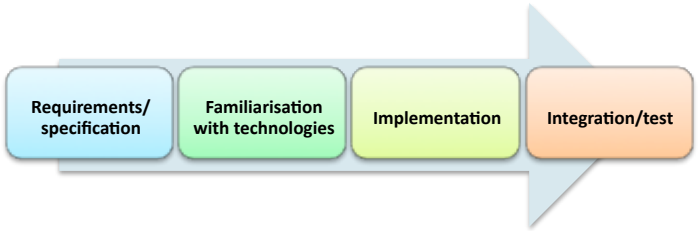

Figure 2.1: Project phases

In course of the project start, a Skype meeting was scheduled on October 22nd. The French and the German team attended this meeting to introduce themselves and to discuss general organizational issues. Due to their geographical closeness to Prof. Laurini, the French team gave a short overview on the system and offered the creation of a very early specification in cooperation with Prof. Laurini in order to provide a more detailed project description.

The follow-up meeting was the most important one of the specification stage since we made most of the organizational and architectural decisions based on the early specification within this meeting. Unfortunately, the Chilean team was not present again, but everybody agreed that there is not enough time left to wait on their first response, in particular due to the pending initial presentation of the German team. The results from this meeting were the determination of the used hardware and technologies and the distribution of the project subtasks. In addition, a raw timetable was scheduled which only consisted of the deadline of the entire project work and the approximately start of integration due to the very low matching of deadlines prescribed by the different universities. Furthermore a weekly meeting, suggested by the German team to improve the communication and synchronization was established for each Thursday.

Shortly after this meeting we had the first contact with the Chilean team in which we briefed them with the recent project state which was not very difficult because all decisions were documented and saved in Google document files, available to each project member. The Chilean team agreed to the vast majority of our decision, so we had only to adjust a few things. The first project stage was that one with the highest communication effort till everything was planned and decided to be able to start the work on the deliverables itself. For the German team, the first stage ended with the preparation of their initial presentation.

Familiarization: Calender Week 46/47 (09.11.2009 - 22.11.2009)
After general decisions for the development had been made and documented, the implementation for each component was prepared in the following two weeks. In case of the German team, that meant the learning of the programming language Objective-C with its common practices and the acquirement of access to Apple computers. Besides the German team applied for the admission in the Apple developer ([Ap09],[St09]) licence of the Brügge chair since only with such a license the application can be installed on the German team's iPhones. The Chilean team had to get familiar with Prof. Laurini's requisites for the image processing and the necessary mathematical foundations for the computation of the arrows. As the team members of France were already familiar with the used technologies, they only had to organize a web server to deploy the application.

Implementation: Calender Week 48/ 49 (23.11 .2009 - 06.12.2009)
In calendar weeks 48 and 49 the French and Germans implemented first versions of their applications. That meant for the French team, that the server application and its web services had to be implemented and deployed on a tomcat web application server in Lyon. The German team developed an early version of the iPhone application with its basic functionality; typing in the address and to download and display pictures. In the next step, the position detection framework and the basic functionality to issue SOAP-requests were integrated in the client. During that stage, there was hardly ever contact with the Chilean team and the French team as our project leaders ensured to intensify the communication effort with the Chilean team. The meetings in this phase were used to adjust and improve some design decisions due to the deepened understanding of technologies in particular on the German side.

Integration/test: Calender Week 50 - 52 (09.12.2009 - 22.12.2009)
After the server application was deployed, the SOAP communication logic of the client had to be adapted to the requisites of the server implementation. Henceforward, the itinerary calculation and the communication between client and server could be tested. During the following two weeks several little changes of the integrated system were implemented, hence the system was improved by adding further functionality such as the calculation and displaying of the distance between the actual position of the pedestrian and the next crossroad. Despite the ensured intervention of the French team, unfortunately only the parts of France and Germany could be integrated and tested during this stage, since the image processing part of Chile was delivered to late. Therefore, the final presentation of the German team could only be held using a prototype system without the image processing part.

2.2.2.2 Problems and challenges

The project started with very low detailed information of the project task. At the beginning it was not really clear what the functional and non-functional requirements of the project are and what should exactly be delivered at the end. Even the French team did not really have much more detailed information which delayed the actual start of the project work unnecessarily. A possible solution could consist of an initial video conference with the responsible Professor to clear out any uncertainty and to set a common point of project start.

Another problem was the difference between the deliverables and deadlines prescribed by each university. This led to difficulties in the creation of presentations and reports. For instance, it nearly happened not to have a working server during the final presentation of the German team since the French team was already on Christmas holidays and wanted to shut down the server due to security reasons. We strongly recommend finding a solution for matching the deadlines in order to prevent a disadvantage of one project team.

In the context of developing a piece of software in an international project team, we had to face different cultures and mentalities that complicated the communication and synchronization among all project members. It needed some time to realize how the participants are used to work. We also had to face the problem of a big time-delay between the central European time (France and Germany) and time zone in Chile of 7 hours. That meant in some cases, a very long reaction time between a request to Chile and the corresponding response. That is maybe one reason why the intensified communication effort with the Chilean team was not as effective as supposed.

2.3 System specification and development

2.3.1 Architecture

During our kick-off meeting, we discussed the possible technologies for the mobile application that we had to develop. We first decided that a standalone client application will not be possible due to the enormous quantity of data that a picture based map implies. So we decided to use a three tier architecture which can be seen in figure 2.2. The server is represented by a web services based application ensuring the interoperability with the client.

2.3.2 Server

The server-side is implemented as a Java web services application, deployed on a Tomcat web application server. We used Eclipse as integrated development environment (IDE) and MySQL as relational database management system. The underlying framework is Maven, which defines the libraries and their dependencies in the "poam.xml" file and finally delivers a web archive.

The database access is managed by the Hibernate framework which maps the entities from its Java classes on the database. Amongst others, this facilitates the modification of the database in case of later extensions.

The other used technologies are the Spring framework for the main singleton and stateless bean components, the Apache CXF framework for the web services registry and soap interface and finally the Eh-Cache framework for the user sessions (with automatic timeouts, idle ...).

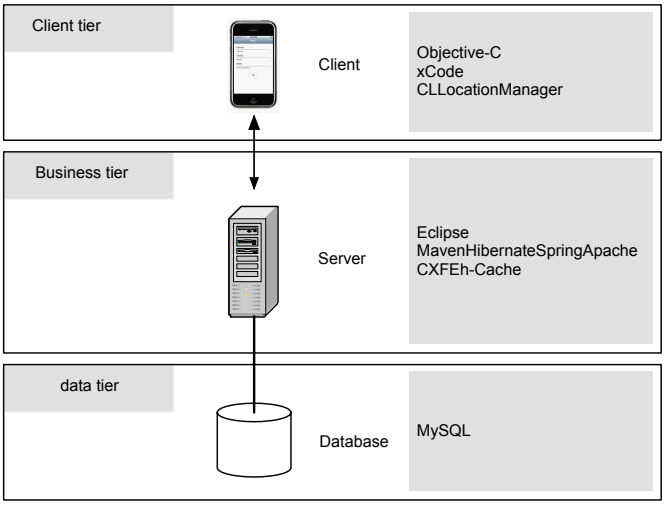

Figure 2.2: Architecture

2.3.2.1 Database

The database schema was realized in MySQL consisting of the entities and relations that can be seen in figure 2.3 and 2.4.

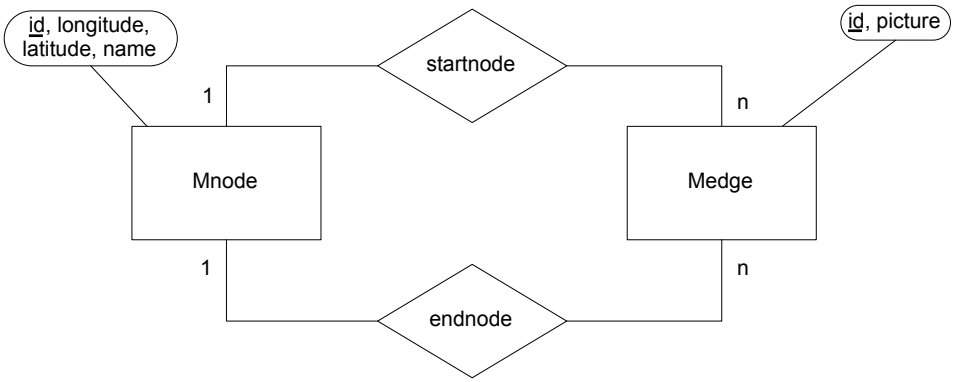

Figure 2.3: Entity relationship (ER) model of the database schema

1. Table Mnode - describes a map node, geo-located by its latitude, longitude, named according to the corresponding address.
2. Table Medge - describes a map edge, oriented by the starting and ending node. Each edge contains the picture taken at a certain distance from the start-node, in the direction of the end-node.

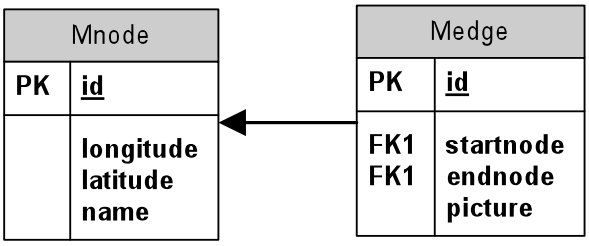

Figure 2.4: Data model in table notation

2.3.2.2 Business logic

When a route calculation is requested by the client-side, an identifier (id) is generated (with numbers and letters) and returned to the client. The id is provided to the server each time a web service operation is called from the client. After a certain period of inactivity, it will be automatically erased on the server.

The communication between server and client is established by using SOAP as remote procedure call protocol.

A map structure is used to associate the generated id with the calculated path, the last client GPS position and the final destination. The picture which the client has to display is always available at the same URL with the following format: `http://server_address/itineraryId.jpg`.

Before the SOAP request is answered, the picture is preprocessed and decorated with an arrow pointing in the direction the pedestrian has to walk. The arrows are calculated against the two following nodes of the route. Depending on this piece of information, the angle of the crossroad is determined by trigonometric functions and used for the arrow calculation. After determining a contrast color, the picture is modified with the arrow and transmitted to the client.

Regarding the main logic for the provision of navigation functionalities, two methods are provided to the client:

- calculateItinerary:

  Client requests the calculateItinerary-method providing the actual GPS coordinates and the final destination. The itinerary is calculated using Dijkstra's shortest path algorithm and stored in a map structure where the key is the generated identifier of the itinerary (itineraryId).

  The method returns the itineraryId if the address exists and throws an exception if not. The exception has the following message: "Final destination cannot be found".

- getNext:

  The Client requests the getNext-method providing the identifier of afore requested itinerary and the changed coordinates. The actual coordinates are verified with the path identified in the map structure by the itineraryId.

  If the itineraryId does not exist in the map structure, the operation throws an exception with the following message: "Cannot find this itinerary. Please check the id". Otherwise considering the client's GPS position one out of six possible states will be returned. Among these states are notifications for the case that the pedestrian goes directly on the right route to the destination. There for the server lets the client know when to change the picture, when the target is reached or when it can be idle. Beside these regular cases the client is notified if the pedestrian is moving in the opposite direction, in a totally wrong direction or out of the map.

2.3.3 Client

For the client device, we had the choice between Apple's iPhone, a Windows Mobile device or an Android device. We decided to use an Apple iPhone as the client device justified by several major reasons: First of all the iPhone provided all required technologies without the requirement of additional devices. Furthermore, the overall performance of the device was superior compared to most competitors. On the software development side, the iPhone offered a comfortable IDE and a lot of important functionalities such as the position detection mechanism, directly out of the standard API. Last but not least, the availability of devices for the German team was another reason for the iPhone. In the context of using an iPhone, we had to develop the client application in Objective-C. As IDE we used the xCode in version 3.1.2 which can only be used on a Macintosh computer.

2.3.3.1 Position-detection

For the position detection, we applied the integrated GPS position detection mechanism of the iPhone and the corresponding Core Location Framework which is offered by the API of the device. After starting the position detection mechanism, this framework provides a CLLocation attribute which enables the determination of the current position. Furthermore the activation provokes a notification each time the actual position has changed. This notification is realized by an automatically called method in whose body the coordinates of the changed position can be readout via CLLocation.latitude and CLLocation.longitude. Within this method also additional functionality can be added, so the getNext-method is invoked here.

2.3.3.2 Communication

To issue the SOAP-calls on the client side, we had to create normal http-requests with the necessary XML-code, because SOAP-calls are not supported by the iPhone API. We distinguished between the invocations of the calculateItinerary-method and the getNext-method. Since these both SOAP-requests only differ in one parameter, we created an own method to handle both requests.

2.3.3.3 Client behavior

The client application consists of two views, the "Home-view" and the "Navigation-view", to display the necessary information (see figure 2.5). The Home-view is displayed when the application is started. Inside this view there are several text fields to type in the destination address. During this process, the actual position is determined using the CLLocationManager of the iPhone's API as already mentioned. Through pressing the "Go-button" the display switches to the Navigation-view and the calculateItinerary-method is called for the first creation of the itinerary. The client transmits the destination address information and the current position and receives the itineraryId.

The Navigation-view is the main view for the navigation process. After it has received the itineraryId from the server, it is able to display the pictures and to validate the position. Every time the current position is updated by the Core Location Framework, we call the getNext-method with the itineraryId and the new position as parameters. How frequently this happens can be adjusted by the accuracy property of the CLLocationManager. Depending on the response of the getNext-Method, we distinguish between the possible cases in order to keep the displayed picture and further information such as the distance to the next crossroad current.

Home-view Navigation-view

Figure 2.5: Home- and Navigationview

2.4 Lessons learned

The first thing we could pull out of the project was that the one of most important tasks in an international team is the synchronization among all participants and stakeholders. The fact that one party continuously was not on the same page as the other project members resulted in an unnecessary communication effort which should be prevented. Our conclusion of that is to establish a well planned communication scheme and a clear report and responsibility hierarchy directly at the beginning of the project.

Furthermore, we saw that it is important to stay in close contact with the customer. That is maybe one of the most important points because only this can ensure to match the requirements of the customer. The lack of this close contact in our specific project could be reasoned in the fact that our customer was more a supervisor rather than a customer with financial intents for the project results.

To sum up the project work and the corresponding experience, it was a complete new experience to work together with other people positioned far away from our own location and to be dependent on these people work. We think that we could increase our personal abilities along with the project progress.

Bibliography

[Ap09] Apple - iPhone dev center:. Website. 2009. http://developer.apple.com/iphone/; visited on December 12th 2009.

[Pr09] Prof. R. Laurini:. 2009. Picture based itineraries - slides.

[St09] Stanford University:. Website. 2009. iPhone application development: http://developer.apple.com/iphone/; visited on December 20th 2009.

CHAPTER 3

Cadaster system

Author:	Martin Fröwis, TUM
Project sponsor:	Prof. Dr. David Sol, Mexico
Participating universities:	Tecnológico de Monterrey campus Puebla (TEC), Mexico Technische Universität München (TUM), Germany Institute National des Sciences Appliquées de Lyon (INSA), France
Team members:	Hector Medina, TEC Oscar Galvéz Fernandez, TEC Ramiro Lopez Lemarrov, TEC Martin Fröwis, TUM Oana Stan, INSA Xavier Faure, INSA

3.1 Introduction

The subject of the project was to provide a system which uses the geographical data of the cadaster system of Puebla, Mexico. The user is able to visualize, create, and modify articles related to a specific location in Puebla. Therefore, the information has to be shown on a map as well as provide a possibility to add new articles which references on locations or geographical points. This information should be arranged in a meaningful method to simplify navigation. The setup can support several applications where spatial data is essential to locate information.

3.1.1 Context

Nowadays, there is a huge amount of available information (80%) which has a geographical component. We can find geographical objects in many domains: urban, design, space management, infrastructure, transport, archeology, etc.. Therefore, more and more applications are using geospatial reference

points. Other phenomenon which is currently taking place is the growth of web 2.0 or semantic web. Moreover, the users are not only reading the web, they are also contributing to the web. Wiki's are the new collaborative environments which allow the construction of knowledge. Users around the world can access Wiki environments and they can add or modify comments and information. One of the most famous wiki environments around the world is Wikipedia. All these wikis are creating communities of users which collaborate together to develop and improve the existing knowledge.

Our project was a GeoWeb application, which helps to identify all the information related to a geographical spot and answer the prototype question: "What is here?". The social context of this project is the progress of the work environment in the computer science field. Due to globalization and increased mobility, there are more and more teams with members of different nationalities. Another practice today is to externalize a part of the work off-shore for reducing costs. In this situation, the employees are forced to work and collaborate at distance.

3.1.2 Challenges

There were many aspects which we had to consider. Due to the distance between Mexico, France and Germany, face to face meetings were impossible. Another issue was that there are 7 hours time difference between Mexico and France so establishing meetings was very difficult, because different schedules had to be considered. Since there were different schools involved, our team members have different educational and personal backgrounds. Therefore we did not have necessary having the same approaches or objectives. Another problem to deal with was that we had to use English as working language, which is not the native language for none of the team members. This led to some misunderstandings during our conception phase. The technical aspects we had to tackle were to find the most appropriate collaborative tools and for some of the members the lack of PHP knowledge.

3.2 Project organization

In order to accomplish this project, we had to find collaboration tools, to divide the roles and to plan the activities, to identify the risks, and track the advancement of the work.

3.2.1 Collaborative tools

For the weekly meetings we have used the video conference systems provided by universities which allowed us better communication by audio-visual. However, because the video conference room was not always available, we had to find another communication means. Our second collaborative tool was Skype, which we used regularly. Caused by some slow Internet connections and the lack of multi-user-meetings in Skype by audio/video, we mainly used text conversations. We have also tried Dim-Dim[1] for one of the meetings. For the work aside the meetings, we used traditional mail for communication between all team members. We have also created a Google group for sharing information between the French team. We used Tortoise SVN as Subversion System, and the server offered by Assembla, which seemed the easiest to manipulate by being free for use.

3.2.2 Division of roles

After discussions between all team members, we have decided to name two project managers:

[1] More information on `http://www.dimdim.com`, last accessed 03/25/10

- One team leader in Mexico, Hector, for managing the Mexican members
- Xavier in France for managing the French team
- Oana was in charge of the documentation, helping the team leaders in writing the reports and being aware of the structure of the writing deliverables
- Responsible of the quality was Oscar which assured that the content corresponds to the requirements and the deliverable type. He verified the text documents for structure/alignment/orthography/grammar
- Martin was the in charge of all the technical aspects. He also verified that all the team members are using correctly the different tools
- Ramiro was responsible of the internal communication assuring that everyone is updated about the project advancement and that the meeting times are respected.

In general, these roles divided just the responsibility of our project. Every phase of the project was divided in separate tasks, which were assigned to specific members of our team independent of their supervisory role. The person in charge was then responsible for coordination, advancement of the tasks and the further integration of the results.

3.2.3 Planning

We have divided our activities in several phases: design, conception, implementation, delivery. Our approach matches to the V-cycle methodology. In Figure 3.1 you can see the Gantt diagram, with the tasks, the duration for each task, and the order they must respect.

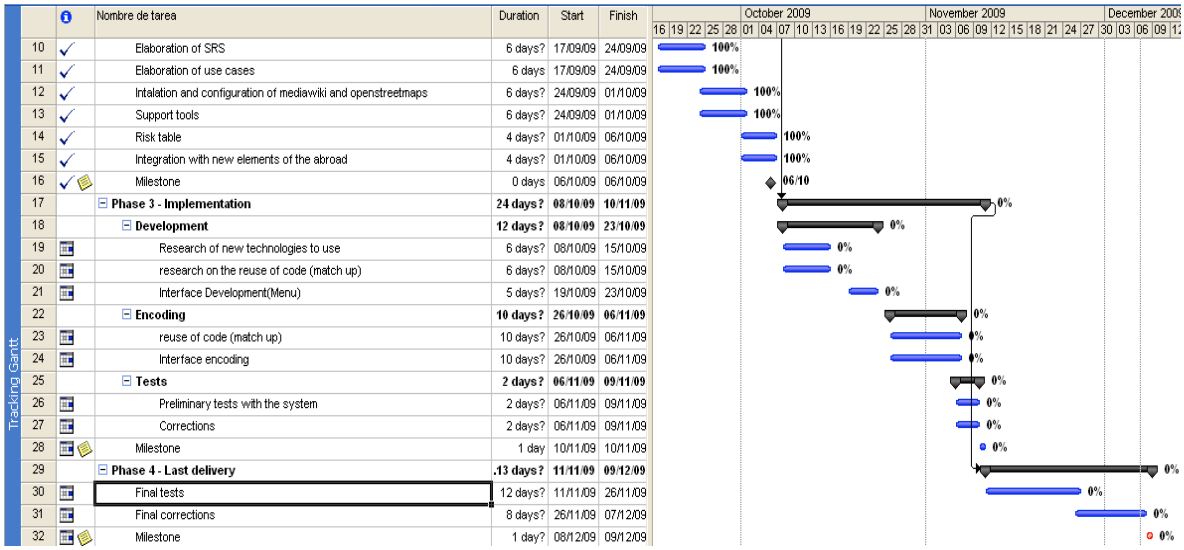

Figure 3.1: Project Plan

3.2.4 Risk identification

In order to avoid risks or minimize the consequences of them, we tried to identify the possible risk factors and the actions to take. We have created a table, where we specified the risk element (if it is the Mexican, French, or German team), the risk description, the probability of occurrence and the

gravity (potential impact). Also we defined which actions should be taken and the costs of response development.

All these risks can be divided into three parts. Generally, the hardware and software problems are the first which people have in mind. It's easy to compile a list of these problems like a server downtime. In this case, we have to wait because we can't access the server physically. But, we can develop on local machines. A conflicted version or a conflict of the used technologies may occur. This risk can be reduced by using subversion.

After the hardware issues, the conception and specification problems were defined. If the dimension and approached of the project are not defined in detail, a team member could misunderstand the subject, work on useless tasks and waste his time. So we organized frequent checkpoints to discuss the problems and define the different tasks in detail, so every team member was clear about the tasks he was supposed to do.

Another problem was the limited time span for the project. If there are time consuming activities which led in the wrong directions we could run out of time. So we decided to assign priority ratings to our tasks. So we could focus on the most essential tasks and work on improvements and detailed adjustments at last.

The specification must be enough detailed, comprehensive, completed, and linked with the conception. There has to be a responsible person for checking the quality of the specification. He / She had to keep in mind these points to avoid problems. An error in specifications would be expensive for the project.

In this project, we used English as working language. We haven't the same easiness to express ourselves, to communicate and to understand this language. So, we have to be more concentrated than during all projects we do before. The team members' availabilities are different. Between Puebla and Lyon/Munich, there are about 6000 miles and 7 hours of time difference. So meetings in the evening in Lyon and Munich were scheduled in the morning for the Mexican members.

We haven't the same education. So, we haven't the same skills. We had to identify the competences of each member of the project to affect tasks depending on the skills of each other.

A member or a person of his family can be ill. In this kind of project, it's difficult to rejoin a team when we missed one work week. The communication is difficult. We never meet the members of the project. It's an obstacle to the comprehension.

So we must quantify, find the impact of the risks. We had to calculate the cost of each risk and find actions to decrease the expenses due to one of these risks (see figure 3.2). But, sometime we had not the time to check this sheet and we applied an emergency reaction.

3.2.5 Tracking the meeting

We kept track of the advancement of our work and the meetings we had with the help of an agenda and meeting minutes. The agenda was prepared before the meetings to assure that the discussion follows certain items marked on it. Also it helped to verify the advancement of work each week and established new action items for the following days. The minutes were edited just after the meeting and were a summary of items which were discussed during the meeting. It was also a way to retrieve information, especially for the members exchanged during a reunion. Minutes were also helpful to find out the date and location of the next meeting.

Risk identification		Risk Quantification		Response development		Risk Control	
WBS Element	Risk Description	Probability of Occurrence	Potential Impact	Action	Cost	Executed Action	Responsible
MTA-1	The availability of the public server (Tec server) is not the desired one	Low	High	Look for another public server and make a back up of the information	Medium	-	-
MTA-1	The public server is down	Low	High	Use a back up server	Medium	-	-
MTA-1	Loss information on the main server	Low	High	Use the back up server	Medium	-	-
MTA-3	Make a wrong design of the index page	Very Low	Medium	Make a design change of the page	Low	-	-
MTA-4	Change core features of the functionality of MW	Medium	High	Make a back up of the information	Low	-	-
MTA-5	Design Time extended	Low	High	Work extra time		-	-
MTA-6	Lack of Cadaster System Support.	Low	Medium	Adopt another map server	High	-	-

Figure 3.2: Risk identification table

3.2.6 Organizational problems

One of our main problems was that the video conference room was not available in Mexico or France, so we had to postpone our meetings or use another communication tool, like Skype. Another problem was the distance and the different time zones. It was not easy to schedule our meetings and presentations so that everyone could join. There were also some personal problems: Xavier got sick and was not able to work. Hector had a funeral in his family.

3.2.7 Possible improvements

We believe that there should be more rooms equipped with video conference systems, especially in France and Mexico. These rooms should be accessible on weekends, too. Also, we would suggest an intermediate review so the work can be tracked and organized (like the German team did). There should be a better communication between the teams too, especially for presentation dates and organizational issues.

3.3 System specification and development

During specification phase, we have established the main characteristics of our product.

First, we decided to develop a GeoWeb application with a map and wiki articles associated to specific locations in the map. The main page explains the aim of the GeoWiki and offers the possibility either to navigate through existing data or to create new articles. The navigation map contains a map to show the different articles and their location. A marker represents the exact location of each article.

All the articles are managed by the users of the GeoWiki. To add one or more articles, a new user has to create an account. There is a search area to find a wiki article and modify it. To create a new document, a user must specify a location from the map, choose a name to the document and post the data to the server. Then, the user will be forwarded to the edit page of his article which already contains a reference corresponding to the position of the new article.

The principle of a wiki is: everybody can modify an article. A user can search an article and select

"edit mode", or do the same searching on the navigation map. A user can modify a wiki article but he cannot delete it before he obtains acceptance of all the community, based on votes. Of course, even if a user is not logged in, he can take a look on the Puebla map and the articles related to a specific location on the map.

3.3.1 Conception

After working out different use cases for our system, we divided our conception into two categories, the technical conception and software conception.

3.3.1.1 Technical Conception

To reach our project goals, we had to combine data from two different sources. First, we used data from our local database to store articles, users, history and permissions. Second, we had to implement an interface to the geographical data, provided by the cadaster system in Puebla. These data is accessible through Web Map Services (WMS). For implementing the graphical user interface and to draw the map, we used OpenLayers. OpenLayers is a JavaScript library for displaying map data in web browsers, with no server-side dependencies. OpenLayers implements a JavaScript API for building rich web-based geographic applications, similar to the Google Maps and MSN Virtual Earth APIs, but it is Open Source.

3.3.1.2 Software Conception

After evaluating different Wiki systems, we decided to use MediaWiki for our implementation. It is Open Source too and allows adoptions by implementing extensions. To show geographical references in our map data, we decided to implement Points of Interests (POIs). These POIs are differentiated in categories like public buildings, police stations, hotels, or restaurants.

3.3.2 Technical aspects

We host our solutions on a server in Puebla running XAMPP. The data of our MediaWiki is therefore stored in a MySQL database. However, during developing phase, we were using local instances of our MediaWiki to keep our independence from Internet connections and hardware issues. To allow users to create articles referenced to a geographical point, we had to adopt the standard MediaWiki. We used an extension called SlippyMap to show cut outs from the map in our articles. Additionally, we developed our own extension, called ScriptButton. This extension is responsible for handling the geographical reference point (by writing latitude and longitude into our database) as well as handling POIs and matching categories. The complete technical environment can be seen in figure 3.3.

3.3.3 Testing

We have done several tests before migrating to a live server. The tests mainly focused on our own extension ScriptButton as well as on the implementation of OpenLayers in our MediaWiki.

We have extended the standard database of MediaWiki with three additional tables which were needed to implement POIs. To avoid problems and possible errors with these tables, we have tested them in several ways, including:

- Creating articles without a title

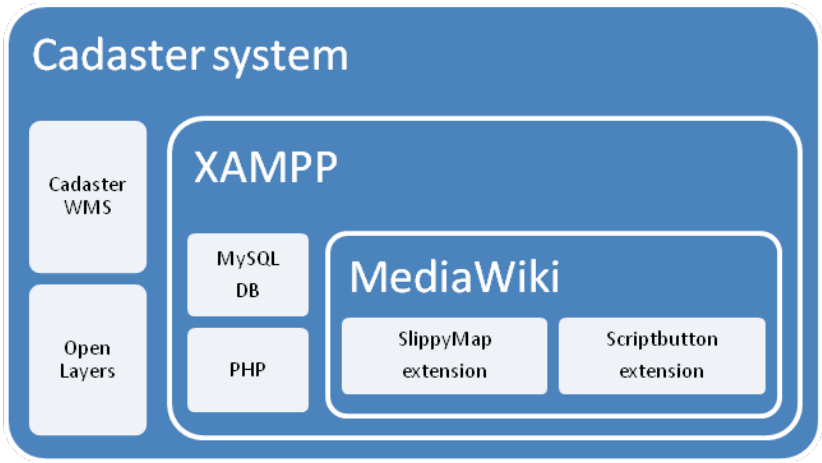

Figure 3.3: Cadaster system - technical environment

- Creating articles without a category
- Creating articles with special characters (which are not supported by MediaWiki)

Additionally, we have tested our implementation of OpenLayers as well as our WMS data:

- Creating articles with reference points out of the region of Puebla
- Creating articles with the same reference points as an existing one
- Creating articles without setting a reference point (e.g. creation on highest zoom level)

We have discovered several minor bugs during our testing procedure which we could fix before migration on a live server.

3.3.4 Technical problems

We had some hardware problems concerning the server in Mexico which hosts our MediaWiki. Therefore we lost some days until the problems were fixed. Caused by Internet connection problems at the campus in France, a voice communication via Skype was not always possible. After the hardware problems on our server, we decided to work on local installations. Therefore, there were a few issues until we got the installations running on every computer, because there were different backgrounds and systems (Windows, Linux and MacOS). But this issue forced us to write detailed manuals to implement our work on other systems, so it turned out to be an advantage as well. We tried to manage all these issues and minimize their effects on the project. Although they affected our productivity, we think that we had learned a lot by solving these problems on our own.

3.3.5 Final product

Our final product is now in the migrating process to a public server in Puebla, which is located and maintained at the Campus of the University of Puebla. In figure 3.4 you see an image of our navigation map.

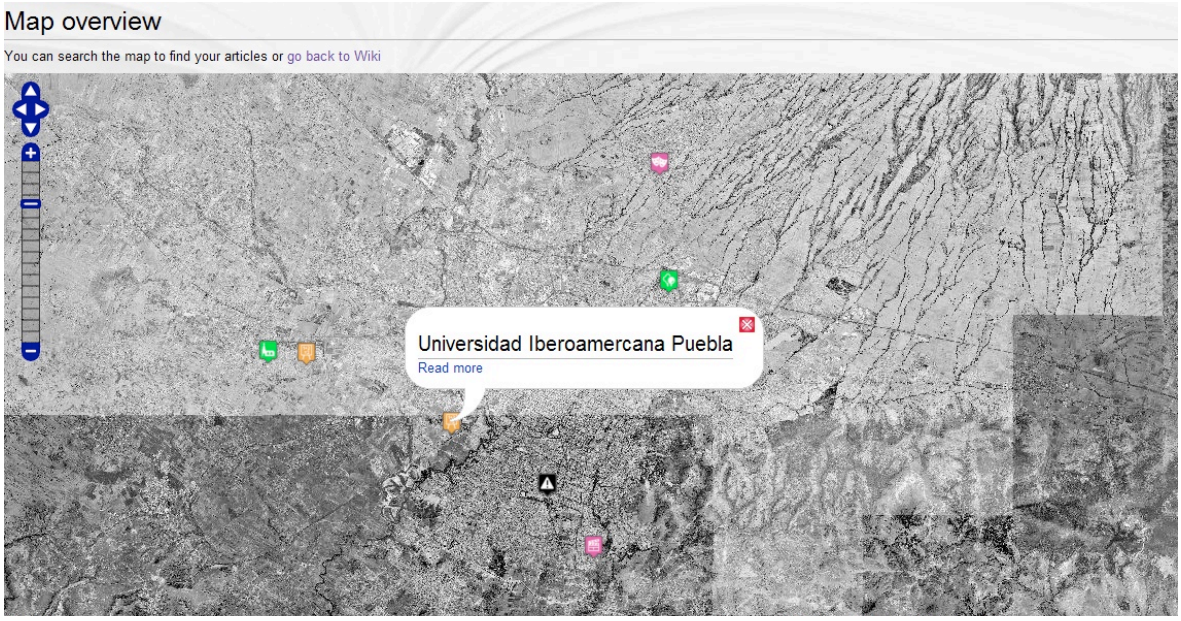

Figure 3.4: Screenshot of the Navigation map

3.3.6 Possible improvements

There are a few issues which could be improved, concerning our cadaster system. There is more detailed data available from the cadaster system in Puebla, but unfortunately it covers just approximately 30% of Puebla at the moment. We would like to switch over to this data as soon as it covers all the area. We also want facilitate to edit or delete geographical reference points in our system. Another point would be to link our POIs to different zoom levels as well as add the possibility to show or hide different categories. Especially if there are hundreds of articles it would be difficult to navigate in the map.

CHAPTER 4

Tricia

Author:	Hicham Souiba
Project sponsor:	Christian Neubert Christopher Schulz
Participating universities:	Tecnológico de Monterrey campus Puebla (TEC), Mexico University of Applied Sciences Esslingen (UE), Germany
Team members:	Carlos Mendoza, TEC Israel Cuautle, TEC Ivan Xolocotzi, TEC Ximena Juarez, UE

4.1 Introduction

TRICIA is an open source knowledge management collaboration tool, and it contains following functionalities:

- Wiki
- Personal & team blogging
- File & directory sharing
- Social Networking
- Security
- Scalability

The conceptual overview of TRICIA can be seen in figure 4.1.

The only missing functionality is internationalization: the application interface is in English, since there are people who do not speak and read English. Therefore, different people, mainly IT students, from different countries (France, Mexico, and Germany) were involved in a project consisting of starting a new module for internationalization. It included developing subsequent items:

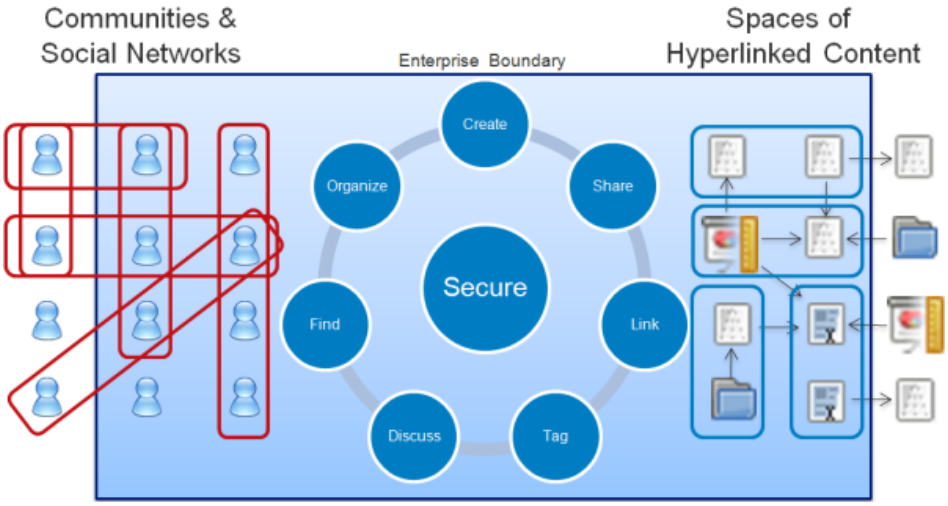

Figure 4.1: Tricia - Conceptual overview

1. Generic import mechanism for language based on Java / XML grammar
2. Runtime language switch
3. Spanish & French translation of TRICIA
4. Test cases.

4.2 Project organization

4.2.1 Context

There were two teams, one at Mexico and another in Europe (Mexican and French), both teams have one leader and there is a global leader. Everybody installed several tools that are: Eclipse, Mercurial, and also we have a schedule and each member have been assigned tasks. Here's a brief description of the used tools:

- Eclipse is a multi-language software development environment comprising an IDE and a plug-in system to extend it. It is written primarily in Java and can be used to develop applications in Java and, by means of the various plug-ins, in other languages as well, including C, C++, COBOL, Python, Perl, and PHP.
- Mercurial is a cross-platform, distributed revision control tool for software developers. It is mainly implemented using the Python programming language, but includes a binary diff implementation written in C. Mercurial was initially written to run on Linux. It has been ported to Windows, Mac OS X, and most other Unix-like systems.
- Dim Dim : Web conferencing that just works, provides easy, open, afforda.
- Skype is a software application that allows users to make voice calls over the Internet. Calls to other users of the service and, in some countries, to free-of-charge numbers, are free, while calls to other landlines and mobile phones can be made for a fee. Additional features include instant messaging, file transfer and video conferencing.

- Windows Live Messenger (formerly named MSN Messenger) is an instant messaging client created by Microsoft.

4.2.2 Work Organization

4.2.2.1 Project Leader and Coordinator

- Hicham Souiba

INSA (Institut National des Sciences Appliquées) of Lyon
Computer Science Department

During the project, the project leader was responsible for:

- Communication with the general customers
- Presenting the general customers wishes to the team
- Coordinate and organize the team work
- Follow up the project progress
- Lead and participate in reviews with the client and / or team
- Realize the wanted technical work.

4.2.2.2 Study group / team

- Carlos Mendoza: TEC, Mexico
- Israel Cuautle: TEC, Mexico
- Ivan Xolocotzi: TEC, Mexico
- Ximena Juarez: UE, Germany

During the project, the project team was responsible for:

- Realize the project technical work.
- Respect team leader constrains (details, follow up, report).

4.2.2.3 General customers representation

- Mr Christian Neubert, TUM
- Mr Christopher Schulz, TUM

4.2.3 Project follow up

4.2.3.1 Follow up rules

- Informal meetings: Meetings between team members were organized regularly to review work progress, answer questions and assign tasks.

- Meeting with the general customers: Four meetings have been hold: Initial presentation done by the general customers team, a follow up meeting and a synchronization meeting done by team members to keep everyone updated of work progress and to ask questions. Finally, a final presentation to present the work that has been performed.

4.3 System specification and development

4.3.1 Approach

As a first step we took a general tour to get acquainted with Tricia. Deployment of this tool gave us the opportunity of know its core functionalities. We understood what Tricia's limitations and functionalities, and the interface it offers to its users.

Currently Tricia shows its controls, accesses, links, and all of its controls of graphical user interface in English. Achieving the first goal of the project (Spanish and French translation) Tricia would be able to show its contents in three languages: English, French, and Spanish. This leads us to create a generic import mechanism for languages based on a Java / XML grammar. Our work on Tricia should be able to change all contents of the graphical user interface to another language, thus more people around the world would enjoy the benefits of Tricia as an Open Source Web Collaboration Tool by using it with their own language. It was asked that our solution would support language switch at runtime. With that users can avoid the necessity of restarting their session or restart Tricia at working time.

Test cases for the import implementation will be the activities running under developer context with objective as guarantee action of importing can be well conducted, without errors. This would not affect principal application performance. Test cases for the language switch implementation will be activities running under developer context as well as the previous goal. It has the objective to check a well translation of all components the user works with, without problems in its visualization. Obviously this task would not cause trouble in the main functionalities of Tricia, because the main issue is to help users to have a good experience with this tool in their own language.

In general, the test cases mentioned previously were not going to touch any other part of source code. Hence, we were obliged to use *jUnit*.

4.3.2 Proposed Solution

Two major goals existed in this project: the first was to implement a generic import mechanism for languages and the second was to enable the language switch at runtime. Other goals were the French and Spanish translation of Tricia and the test cases.

4.3.2.1 Import Mechanism

For the generic import mechanism, the development team used the export mechanism that already existed. This export mechanism provided a XML file, messagesForTranslation.xml, which contained all the existent messages and its translations. Therefore, there were also two cases for the import, either one or all the plugins. However, the process was the same for both cases. Only a condition was added to check whenever a specific plugin was given and then import just the specified plugin. An *Ant* task was created to handle each case. The resulting class name is ***ImportFromTranslation***, located in the *de.infoasset.platform.services.internationalization.scripts* package.

The export mechanism yielded a XML file which was read with *dom4j*. This Java library helps one to read any XML file from a Java source code using the XML structure. The data obtained from the XML file were: the plugin name, the keys of the languages that were in the translation, and all the messages with the translations. Also the extension of the *Messages.** was obtained in order to know the type of file that needed to be modified.

Depending on the extension, the method to change each type of file was called, but the underlying process remained the same. This process was to read the existing *Messages.**, get the existing messages and translations, add the new translations, replace the old ones with the new ones, and at last write the new file. Reading the existent *Messages.** was necessary in order to avoid deleting translations that existed in the *Messages.**, but were not in messagesForTranslation.xml, because the translation had been added after exporting the messages.

For the XML file, the reading and writing was done with *dom4j*. Firstly, one had to get the column that corresponded to each language in case the language exists, or add it otherwise. Then the messages translations were added in each corresponding place or new cells were added with *dom4j*. Finally, the file was written with the ***XMLWriter*** class. The path to where to write each file was obtained with the ***Utilities*** class already provided.

For the Java file, the reading was done using the concept of reflections. In this case, other types of messages had to be considered, namely the ***ParameterizedMessage*** class. A list with the message keys of this type was created. After getting all the messages with its translations in a proper data structure, the keys of the languages were obtained too. In addition, the package name for the Java class, acquired with the ***Utilities*** and ***PackageMarker*** classes, was required.

Taking into account the above discussed elements, the new Java class was easily written. The messages had to be modified to skip the double quote and backslash. Otherwise an exception would be thrown. As previously said, an *Ant* task was written to handle each case: Import the translations of one plugin or for all of them. These tasks were written in the build.xml file located in the *toro* plugin.

4.3.2.2 Language switch

Once we have the translations, we needed to provide the user an appropriated interface to choose his preferred language and a mechanism to deal with this request. Therefore, the default template was changed and a new handler was written. The default template located in the *jakob* plugin was modified to reflect the available languages for Tricia. Each language is represented with a flag icon. This icon has a link that redirects the user to the home page of Tricia in the selected language. The icons of each language were added in the icons folder of the *jakob* plugin. The filename of each icon is the language key. The language key is the lowercase two-letter ISO-639 code. To operate the substitution, we use the Tricia default template substitution system (***PrintSubstitution, ListSubstitution***).

In order to provide the correct functionality in the template, three new methods were written. These functions were located in the Function.java file in the *toro* plugin. These functions give information about the available languages. The function *available_languages* returns a PrintSubstitution which prints the language of the local session. This function is used for test reasons only. The function *count_languages* returns how many languages are available in each instance. It was implemented mainly for debugging purposes and was used in the default template. The function *availableLanguages* returns a ***ListSubstitution*** that is used to iterate on the available languages. This function is used in the template to get the appropriated icon, URL, name and key of each language. Each language item in the list has an id, the key, a name, and a Boolean indicating if it is the current language or not.

When a user clicks on a flag icon, a new request is generated and the language handler is called. This handler inherits from the abstract class ***Handler*** and implements the *NoTestRecording* interface.

From the Handler class it overwrites the method *doBussinesLogic*. In this method the value of the parameter lang of the request is obtained and set as current language in the local session. Next, the request is forwarded to the home page, translated to the selected language. This mechanism can be seen in figure 4.2.

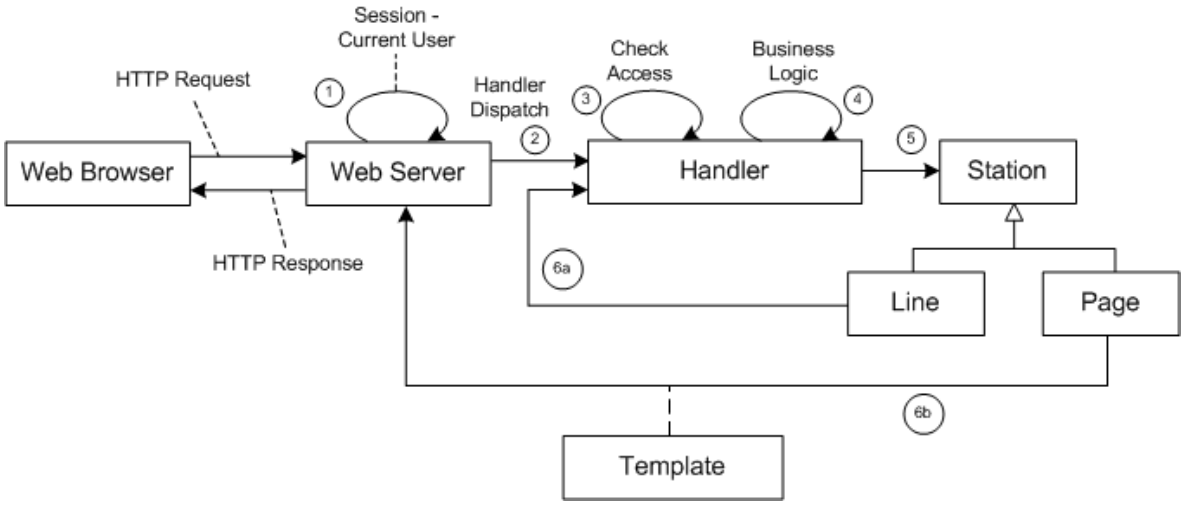

Figure 4.2: Tricia Handler Mechanism

4.3.2.3 Test cases

The test cases were written for the import mechanism only. For this task *JUnit* was used. This library allowed us to write test for a class in a simplified way. Tests are used to check if a class behaves in the expected way by comparing the expected result with the one obtained. There are two things that can be tested: the returning value and the side-effects.

For the import mechanism the result was the correct Java and XML files. But to be sure that those were the correct files one had to obtain them first. This made the writing of the tests recursive. Therefore, just one test was written to try out if any exceptions were thrown. In this test the static method *readTranslationXML* was called to test just one plugin.

4.3.3 Language translation

The development team consisted of persons of different countries, speakers of different languages. This cultural differences were used in the project to obtain the translate Tricia to French and Spanish. As native speakers and frequent Internet users, the members of the development team were capable to provide a trustworthy and accurate translation of all the messages for all the plugins. These translations are available in the messagesForTranslation.xml file.

4.4 Lessons learned

A project involving the work with international clients and international partners requires a lot of effort and good planning. The overall goal of adding new functionalities was interesting although working with projects in process brings some challenges. First you need to adapt to the characteristics

of it, that includes the programming language, classes or methods used and mainly you need to adapt to a new organization and planning scheme. On the other hand it makes easier the fast growth of it because you can add people involved in different areas to add their personal knowledge and make the project more complex and global which is the case of Tricia because we as different language speakers added a new feature to the system, the possibility to interact with it in different languages.

During this project, we learned many things as a team and individually. For the majority of us it was our first international project and that changed the traditional way we have been doing things so far. It provided us a new vision of how it is to work and interact with people from other countries who have different cultures.

Technically, the internationalization and localization of Tricia provided us a lot of new learning that included the use of new tools and the enhancement of some. Tricia was a new tool for us, it has a lot of functionalities and because of its size and its importance it helped us a lot to get used to working with more complex and bigger software. Also, in the technical sense we learned Mercurial which is an important source control management tool really helpful for working with versioned files. Mercurial can help us a lot in further projects, international and local ones. Eclipse which was a familiar development tool to us became really useful and necessary in the process and development of the project. We learned how to manage a huge amount of plugins, simplify the way to interact with classes, how to handle documentation using the Javadoc in eclipse and finally how to obtain the maximum benefit of a development tool which is now really useful and provide a more organized and well planned coding. XML has become a popular language to exchange a huge amount of data within different platforms. In our case with the interaction of XML using Java, we learned a new library: dom4j. With this open source library you can parse an XML file and use it in Java, in our case it was necessary for working with the translation and parsing of the XML files in the project.

By working in a group, we also learned a lot of new things from organization to how to handle communication with our clients and partners. In the organization part we learned the use of new tools such as doodle that was really helpful for all of us to have some time together for the meetings and presentations. We created a Google group to exchange mails in an easier way and to share files. DimDim was a new tool for some of us and represents a complete tool such as Webex, the difference is that DimDim is free.

Finally the most important thing about this project related to the working experience was the international aspect. To have partners with different time zones and different languages was a complete challenge but at the end it provided us a lot of new and very important knowledge.

CHAPTER 5

University relations map

Author:	Sebastian Krebs, TUM Stephan Krusche, TUM
Project sponsor:	Prof. Dr. Florian Matthes, Germany
Participating universities:	Technische Universität München (TUM), Germany University of Applied Sciences Esslingen (UE), Germany Institute National des Sciences Appliquées de Lyon (INSA), France
Team members:	Stephan Krusche, TUM Sebastian Krebs, TUM Sergio Hinojosa, UE Marco Nahm, UE Tanh Nguyen Hoang, INSA Gishlein Thau, INSA

5.1 Introduction

5.1.1 Project idea

Student exchange programs like for example Erasmus [1] gained great popularity over the last years. The numbers of studying people abroad joining the Erasmus program doubles within the last ten years which is illustrated in figure 5.1.

The number of relationships between universities has also increased significantly in the last few years. More than 100 of currently 153 partner universities relations of the Technische Universität München were created in the last decade whereas before only about 50 partners were included in 25 years.[2] This

[1] More information on `http://ec.europa.eu/education/lifelong-learning-programme/doc80_en.htm`, visited on January 18th 2010

[2] Details can be found under `http://portal.mytum.de/international/kooperationen/partneruniversitaeten/index_html`, visited on January 18th 2010

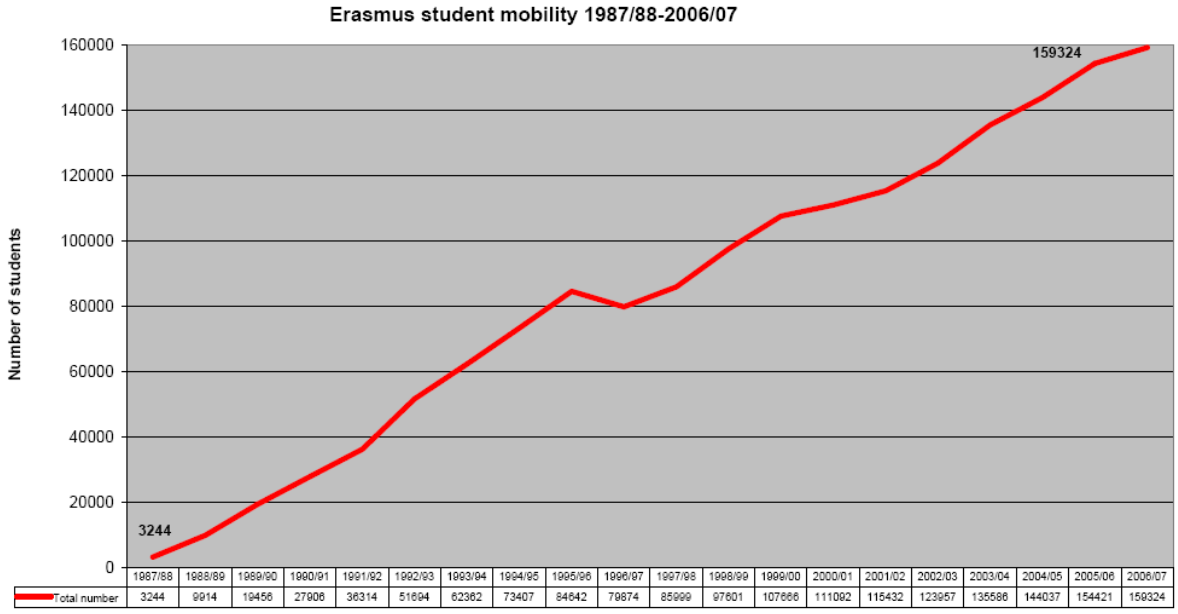

Figure 5.1: Number of students who study abroad with Erasmus

dynamic progress which evolves at every university in the world is not supported yet by any websites or software system.

Students who want to join a study abroad have to deal with different university websites and their own system of showing partnerships. Sometimes they are also faced with the problem that specific faculties of an university do not have a website with information. Therefore, hours of frustrating online research are needed. Another problem occurs when students want to see ratings about their possible studying place or if they want to know more about the possibilities and problems of previous exchange students' lives.

All these problems were the reason for Professor Warendorf from "Esslingen University of Applied Sciences, Germany" (UE[3]) to initiate the project "University foreign relations map". It was embedded in a course called "Software engineering in international teams". The main goal of this course is to *"cooperate with other students from international universities by commonly working on a software engineering project during a period of approximately 2 months"* [So10]. The practical course is embedded in the NEREID (Network of Engineering univeRsities Educating in Intercultural Design [NE10]) cooperation between the computer departments of five different universities in France, Chile, Mexico, and Germany.

Prof. Warendorf asked for a prototype of a greenfield developed system which consists of the following three major parts:

1. A start page with a map for a quick overview on all partner universities and with information about the selected university.
2. An entry mask user interface for academics to administrate information and relationships between universities and to support an easy data upload of pictures.
3. A forum to enable communication between students and uploading of field reports for a specific institution.

[3]See `http://www.hs-esslingen.de/en/44`, visited on January 18th 2010

These three major subsystems had be developed to eliminate the previous mentioned problems and should be integrated via one easy to use website with a simple navigation scheme. All the data should be stored in one database to avoid double entries of information or inconsistencies. *"The goal of this project is to study how an interactive map (Google Maps) can be established to store/link to pertinent information"* [Un10]. The development will lead to a student exchange information system, where students can find partner universities and information about them. It will also be possible to contact foreign students and to look up experience reports of them.

5.2 Project organization

5.2.1 Project organization

The desired system cannot be developed until the project management has set up the project environment and decided how to assign work tasks to teams. This section provides an overview about the chosen software development process and about the project structure.

5.2.1.1 Project plan

The project started at the beginning of October 2009 and had a duration of only ten weeks. It consisted of overlapping phases as it is illustrated on the project plan in figure 5.2.

	Jobs	Start	End	Term
1	Server running	01.10.2009	28.10.2009	4w
2	Team set	01.10.2009	28.10.2009	4w
3	Specification	05.10.2009	17.11.2009	6w 2t
4	Implementation	29.10.2009	18.12.2009	7w 2t
5	Testing	16.11.2009	18.12.2009	5w
6	Presentation	14.12.2009	18.12.2009	1w
7	Documentation	01.10.2009	18.01.2010	15w 3t

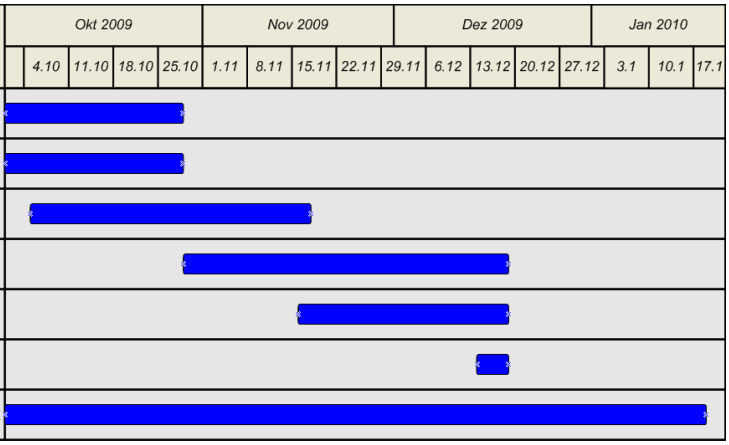

Figure 5.2: Project plan

The project leader did not choose a specific software process or methodology. Therefore the plan looks like the waterfall model. Instead a more practical iterative process was pursued by the participating students. The implementation started before specification was finished because not all requirements were really clear at the beginning of coding. On the other side the probability that requirements will change was very high, as it was a greenfield project. The due date of the delivery of the implemented system was set to the 18th of December 2009.

5.2.1.2 Project structure

To support the exchange and collaboration between universities from different countries the system should be created within an international software engineering project. The third participating univer-

sity besides UE and TUM was "Institut National des Sciences Appliquées de Lyon, France" (INSA[4]) Institut National des Sciences Appliquées de Lyon, which enabled the project to be international. Two students attended the venture from all three universities. An overview about the structure of the project with all team members is shown in figure 5.3. Christopher Schulz and Christian Neubert, who are not shown in the picture, helped as additional tutors on the part of the TUM students.

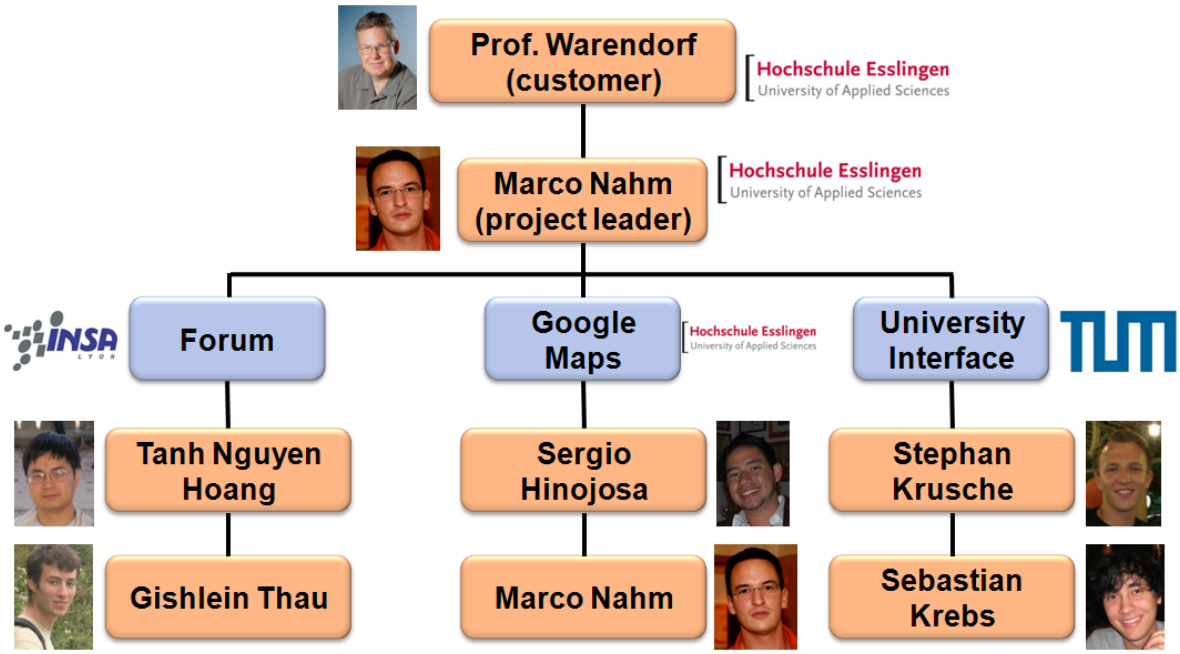

Figure 5.3: Overview of the organisational structure

Professor Warendorf was customer of the project. Marco Nahm was his project leader and divided the team according to the three main requirements into three sub teams. He came to this decision to reduce communication overhead. Each sub team could concentrate on their part of work with local meetings while communication between them concerned only the interfaces between the three work packages. A weekly Skype meeting was scheduled to share problems and information between all participants. The project members had not much time to implement the mentioned system by the reason that they had other courses at their universities. Therefore they could not work full time on this project.

5.2.1.3 Risk management

Although a calculation of possible problems should be part of every project, the project leader surprisingly did not try to initiate any kind of risk management process. Nevertheless, the students from TUM made an approximate risk analysis and identified that - as largely mentioned by [Se07] in the "top ten list of risks in software engineering" - personal shortfalls were the most threatening danger. As the team consisted of only six people, it was clear that any loss of workforce would lead to serious trouble for the rest of the group.

In order to minimize this risk, it was decided to hold regular Skype meetings and stay in contact with each other. By this means, it was thought to be more unlikely that one co-worker felt excluded of the project team, which probably would result in a loss of workforce.

[4]See `http://www.insa-lyon.fr`, visited on January 18th 2010

Furthermore, the project team was aware of another risk, which can be called unrealistic schedules and missing time management. The short time of two months and ten days for this greenfield engineering project turned out to be a big challenge for the small team. For this reason, it was decided that integration should start early in order to have a running prototype at any time for demonstration purposes.

The risks of communication overhead or a lack of work synchronization, which could also compromise the project, were reduced by the separation into three sub teams. Moving targets regarding changing requirements were expected with a high probability. This hazard was minimized by overlapping phases of specification and implementation.

Of course, other risks like complicated communication - which may apply especially to an international distributed development because of different languages - were also anticipated, but these were included in the two top risks mentioned before.

5.2.2 Problems and project challenges

Each software developing team had to face a various number of problems which occurred in the course of the project. There were some problems that have been accepted because their solving had cost too much resources. In contrast, others existed which have been solved immediately since they were likely to result in a complete project failure. The following section describes several problems which occurred during the progress of the project, although an approximate risk analysis and risk mitigation arrangements were made at the beginning.

Organizational problems were mainly caused by the fact, that the three different universities INSA, UE and TUM had different levels of interest in the outcome of the project. After the first three weeks it was obvious - according to working output and motivation shown in the Skype meetings - that the course at INSA was not regarded as being important. In contrast the students from UE and TUM showed interest in the success of the project, although the sub team from UE hold the view that, as long as the application was running, it would be good enough - no matter if quality goals were accomplished or not.

Besides that, the customer - Prof. Warendorf - from UE seemed to be too busy to provide the specific and detailed requirements and feedback which were needed. This became clear, when it came to the requirement analysis. Prof. Warendorf therefore sent an e-mail in which he gave some basic instructions - like the contact data of the co-workers - and advised the students to contact the project leader. The project leader in turn remarked, that in such a small project a requirement analysis is not really necessary. Consequently there were some tries to create a requirement document, but these documents were incomplete and not used during the rest of the project. Due to this fact, the students from TUM sometimes did not really know if a certain function was needed or not, which can be dangerous, because according to "Glass' Law" [Gl98] insufficient requirements are the primary source of project failures. Luckily for the team Glass Law did not apply to the NEREID project.

The lack of motivation of the French side resulted in a serious problem for the team, because their team members - who were responsible for the forum - did not deliver the desired output in time. Until a few weeks before the end of the project the rest of the team was not aware of the circumstance, that the students from INSA did not really have the intention to contribute their part to the project, using delay tactics like telling that they have no time or serious problems at the moment and have to postpone the check-in. Ex post it is obvious that the an earlier escalation of the situation had prevented at least the uncertainness factor. In addition a higher amount of project experience among the team members in general may have avoided this situation.

5.3 System specification and development

5.3.1 Solutions

This chapter provides the developed solutions of the university map including the requirements as well as the specification of the system and a description how it was developed. The first important activity in the project was to evaluate possible technologies which can be used to develop and implement an university map on the Internet. This was early done and the following decisions were made.

5.3.1.1 Technology

To work together on the same files the open source tool subversion (SVN[5]) Subversion was used as configuration management software. The subversion repository was provided by UE and is reachable under `https://svn.hs-esslingen.de:443/~warendor/svn/NEREID`. Because of existing solutions for the forum and an easy integration of Google Maps, Active Server Pages .NET[6] (ASP.NET) Active Server Pages .NET with C# as programming language was the tool for the website development.

That enabled the simple utilization of Asynchronous JavaScript and eXtensible Markup Language (AJAX and XML) which allowed to create Internet pages that dynamically load their content without refreshing a whole page. A better usability for users was guaranteed, because the response time for selecting another university and loading its data into the website or uploading pictures into the database is very low. Another reason for this technology was the opinion of the project leader that it is easy to use, to develop with it, and to deploy the system on a web server. In the end, this fact was proved wrong when a lot of problems occurred during the late integration of the system.

ASP.NET requires a Microsoft SQL Structured Query Language Server if as database connection an object relational mapping (ORM) Object relational mapping will be used. This was decided by the TUM sub team because language integrated queries (LINQ)[7] can be used instead of SQL strings. The advantage of LINQ is that the compiler can check the database queries for errors.

The Microsoft SQL Server 2008 enterprise edition was used because it is provided by Maniac[8] at no charge. Visual Studio 2008 was used as integrated development environment (IDE) Integrated Development Environment as it is the only tool for development with ASP.NET. It was also free of charge for the participated students since the tool was supplied by Maniac, too.

5.3.1.2 Use cases

Before starting with implementation, good software engineers have always to make requirements elicitation and analysis to describe the full system as the customer wants it. There are many possibilities to model requirements and systems, but the most common is to use the Unified Modeling Language (UML). The functional model, represented in UML with use case diagrams, describes the functionality of the system from the user's point of view [BD09].

As mentioned in chapter 5.1.1, the university map system consists of three main subsystems. Therefore it would be reasonable to design three use case diagrams. But as the forum has not really new functionality and an existing framework called YAF[9] (Yet Another Forum) was used, the use cases are

[5] See `http://tortoisesvn.tigris.org`, visited on January 18th 2010
[6] See `http://www.asp.net`, visited on January 18th 2010
[7] For further information about LINQ see `http://msdn.microsoft.com/en-us/library/bb425822.aspx`, visited on January 18th 2010, and [PP08] Language INtegrated Queries
[8] See for example `http://maniacweb.informatik.tu-muenchen.de`, visited on January 18th 2010
[9] See `http://www.yetanotherforum.net/`, visited on January 18th 2010

actually not interesting. Thus, only the uses cases for the portal and for the database interface were modeled. The first diagram is shown in figure 5.4.

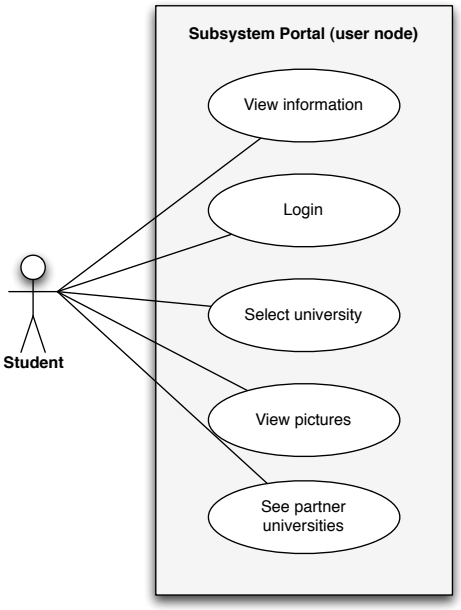

Figure 5.4: Use case diagram of the start page

The main user of the start page will be a student who wants to obtain a first impression of an university. Visiting the university map he or she can look up information, select other universities, view pictures, and see all partner universities in the Google Map. Perhaps the student has also some administration rights for inserting data or updating information of an university. Then he or she can also login and will be directed to the entry mask of the user interface.

Only if the login succeeds, the administrator has the rights do edit every university which is managed by him. A login is also possible from the user interface website. But if the login fails because of a not existing user name or wrong password the person can do nothing on this site.

In editing mode the administrator can change the information which is shown on the start site like the short or long description. He is also able to add new pictures or delete old ones. It is possible to add new partner universities or delete existing ones, too. An administrator has also the possibility to add new universities or faculties as well as to delete existing ones if he / she has the rights. An overview of all use cases can be seen in the diagram in figure 5.5.

5.3.1.3 Design goals

The main design goals of the university foreign relation map system came from the non functional requirements. They were prioritized by the whole team before the design phase took place by respecting subsequent order:

- Usability
- Extensibility
- Quick overview
- Consistency

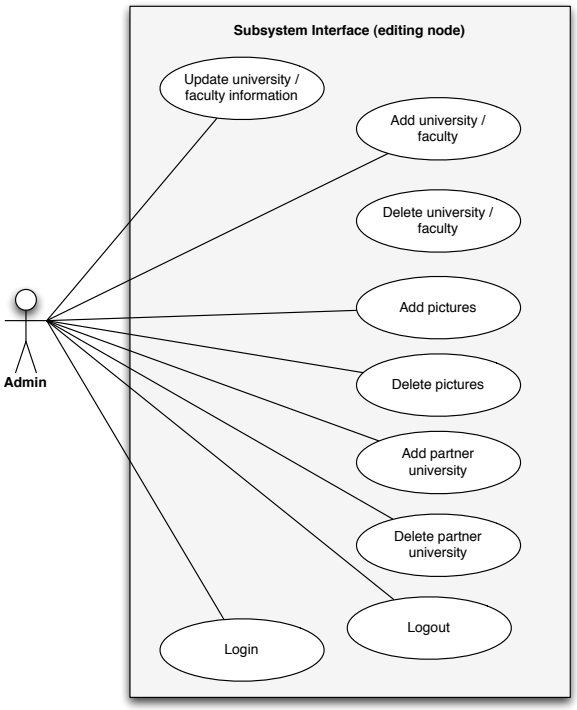

Figure 5.5: Use case diagram of the entry mask user interface

- Integrity

To achieve this goals during this project it was important to create an easy to use and intuitive graphical user interface. The different websites should appear similar and it should be possible to navigate with only few clicks.

Extensibility is *"the quality of a system indicating how easily the system can be changed to accommodate new functionality"* [BD09, page 765]. Extensibility can be reached by intelligent system decomposition with focus on minimizing dependencies between critical classes which might change in the future. *"Coupling measures the dependencies between two subsystems, whereas cohesion measures the dependencies among classes within a subsystem. Ideal subsystem decomposition should minimize coupling and maximize cohesion"* [BD09, page 262].

Other goals like consistency and integrity leads to the decisions to use a database with object mapping and to create methods which prove this before changing the database tables.

5.3.1.4 Subsystem decomposition

The mentioned design goals influenced the separation of the university system into different components. Due to the possible reuse and the wish to upgrade the system in the future, it is not developed "all in one class". The design is easy to extend because the data connection is separated from the two views. The simplified overview on the different parts of the system is shown in figure 5.6.

Both subsystems portal and interface delegate database queries to the class *DataConnection* where all LINQ statements are saved. If the database schema will change in the future because of new requirements, only the subsystem data has to be adapted. The class *ObjectMapping* inside this component is

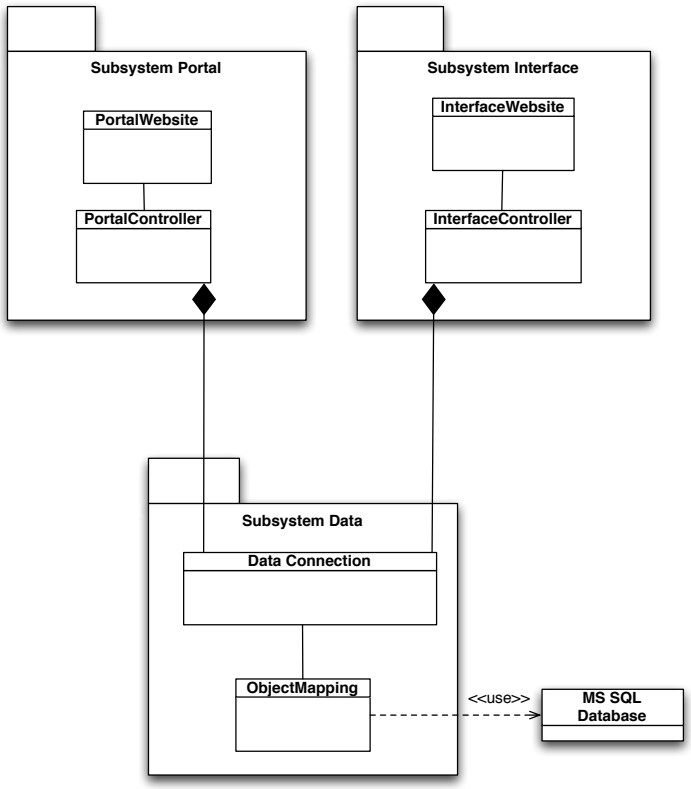

Figure 5.6: Simplified subsystem overview

automatically generated from the database schema which will be explained in the next section 5.3.1.5, so only the class *DataConnection* has to be changed.

That is also important since ASP.NET is used for the classes *PortalWebsite* and *InterfaceWebsite*. Both files are written in XHTML (Extensible Hypertext Markup Language) eXtensible HyperText Markup Language. There the whole graphical user interface is described with ASP extensions. The two controller classes are written in C# and contain all events which will be fired at user actions like a click on a certain button. Then the delegate method with a LINQ query from the class *DataConnection* will be executed to get the desired response.

5.3.1.5 Database schema

Each table in the database also exists as an object inside the *ObjectMapping* class. The tables inside the database are relatively simply structured. A university can consist of several faculties which may have different degree programs. Every entity of these three is administrated by a person. Important is the fact that persons can manage more than one university or faculty.

Faculties and universities have addresses which are important for the icon on the Google Map as well as pictures which are shown on the start page. It is possible that different faculties can have the same address if they are not distributed. The table partnership was created to model the many to many relationship between universities. Thus partnerships between them are possible, which is important to show these connections to users on the Google Map. Reflexive relations between universities can be inserted into the database but cannot be added on the website since the system only showed other universities to add as partner university.

The database schema including the column names and primary unique keys which should be self explanatory is shown in figure 5.7 (screenshot from the Microsoft SQL Server Management Studio). With a Visual Studio 2008 command prompt it is possible to generate automatically the LINQ to SQL object mapping which is shown in figure 5.6 as class ObjectMapping. Each table from the database corresponds to an object. Columns of the table will be inserted as attributes in this object. Relations between tables are mapped to associations of objects. These informations are stored inside the DBML (DataBase Markup Language) file which is in fact an XML file. Visual Studio can now generate classes and associations from this *"NereidUniversityData.dbml"* automatically if it is added to the solution.

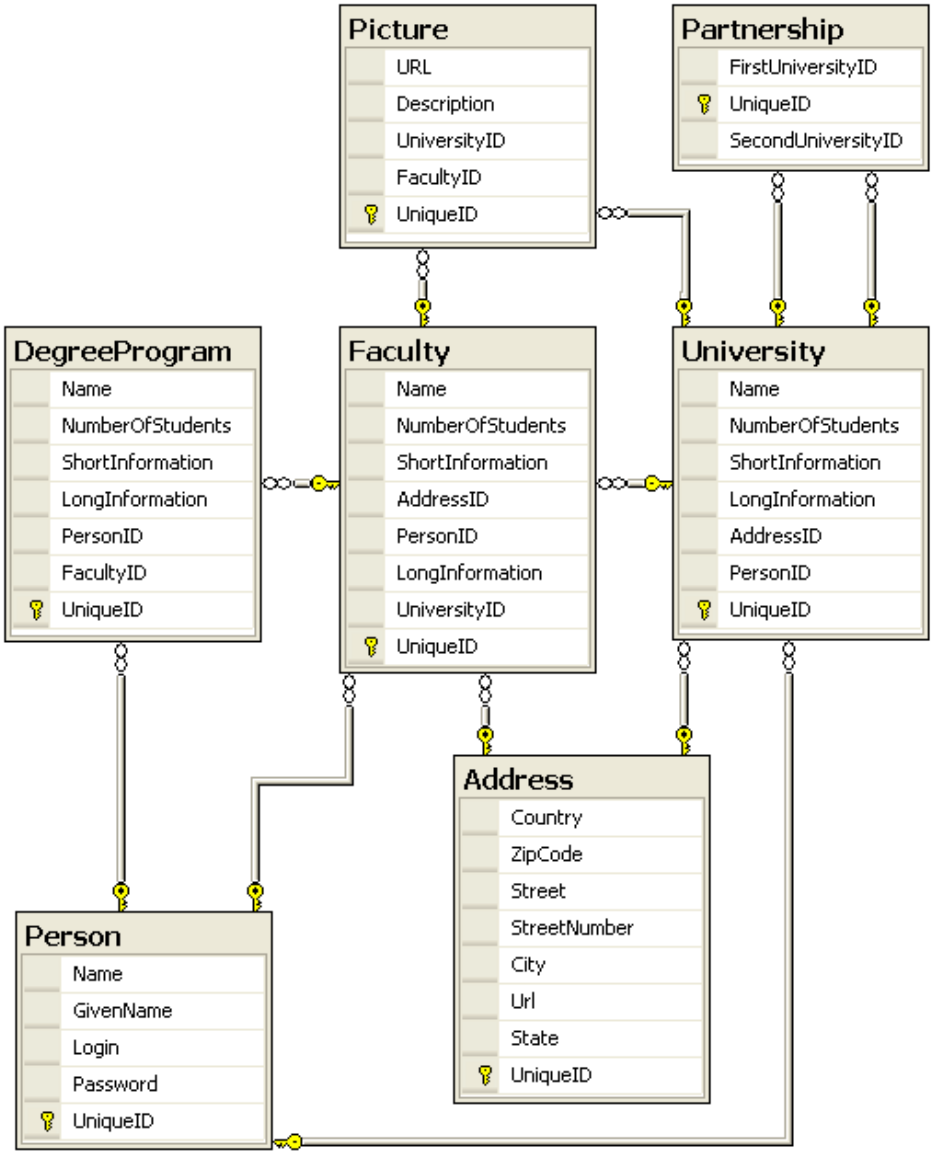

Figure 5.7: Database model

When a database connection will be established at the start of the system every row of all tables will be created as the corresponding object. Finally with the LINQ statements objects will be generated, changed, or deleted at runtime. These objects will then be mapped back to rows in their corresponding table when the changes are submitted to the database.

5.3.1.6 Deployment

After some problems which will be explained in detail in section 5.3.2.1 the developed website was successfully deployed to a web server from UE. This was done shortly before the final presentation of the TUM which was delayed to the 22nd of December 2009. The system can be accessed on `http://nereid.hs-esslingen.de` and is usable since this day.

Every requirement except for the use of only one database for university data and forum was fulfilled. The reason is that the French sub team did not complete their work which will be elaborated on section 5.2.2. Therefore it was not enough time for the other people to customize the existing forum which was implemented in ASP.NET to use the existing Microsoft SQL database server. But this adaption should not be a problem if it is really wanted by the customer in the future.

5.3.2 Problems

5.3.2.1 Technical problems

Due to the choice of technology, some students had difficulties to get the IDE called Visual Studio 2008. The students from INSA did not have access to a license and thus had to use a free trial version while the sub team from TUM as well as the students from UE were capable to work with the full license. But UE had to use Visual Studio 2010 on the server side. This led to some time consuming complications, as the version of 2010 creates files in the building process which are not compatible with the IDE of 2008.

The other serious technical problem was, that only the two students from UE got access granted by the server management group of UE. Thus, it was not possible for the teammates to tackle problems with the server, which turned out to be especially the case for the rights management of the Microsoft SQL Server Management Studio. It took an unexpected amount of time to handle these problems and to get familiar with the AJAX and the ASP.NET technology including C# and LINQ to SQL.

5.3.2.2 Development problems

As only UE had access to the deployment server, the other co-workers from INSA and TUM had no possibility to start integration of the application and the database. Although the students from TUM insisted on an early integration, the database setup and the integration of the application on the server was done by UE only some days before the final report. Therefore the application could only be tested and debugged on the local machines, not on the server. Because of that it can happen, that some bugs - due to the interaction of the application with the server environment - are still undetected.

Another challenge for the students from TUM was the handling of the SVN repository. Large check-ins were made only about every three or four weeks by the team from UE, and thus it was quite time consuming to work with the code. It even occurred, that - without any comments in the SVN log - check-ins of code were made which did not even compile. This made synchronization more difficult than it could have been.

Generally documentation was not estimated as being relevant. For example, the Skype minutes were only written down by the students from TUM at the beginning of the project, and not when it was other co-workers' turn. Moreover, the code which was checked in into the SVN, was often not only uncommented in the SVN log, explaining comments were missing within the code itself, too.

Overall, the project was not as fraught with problems as the reader might assume after having read the last section. Considering the results, the teamwork, especially between the students from TUM was very productive. Finally the positive effects of errors must not be forgotten - the learning effects.

5.4 Lessons learned

Though the two most threatening problems occurred during the project progress, the team was capable to set up a running version of the website at `http://www.nereid.hs-esslingen.de`. After all, the problems which came up and had to be handled, are well-known in the field of software engineering. However, there is of course a big difference between theory and praxis and most of the lessons have only been learned because of the mistakes mentioned in the section before. The reader now might have a too negative impression of the project, having in mind all the things that went wrong. But he or she has to perceive, that all these flaws contribute to the actual aim of this course, the learning about software projects in international teams. The following subsections provide a short overview on the newly gained knowledge and realized improvements:

5.4.1 Project organization

Concerning the organization, one can tell that it is very important to detect conflict potential as early as possible and be prepared for an escalation. However, it is still hard to say - especially without having much software project experience - whether a conflict escalation is a silver bullet in this kind of situation. In a scenario like the NEREID project, the project leader undoubtedly has to show more presence and be stricter with his or her co-workers, although it is hard to build up pressure in an international project without incentives or penalties. That is because the employees did not get any salary for their work, and so the project leader could hardly come up with financial threats. Vice versa, if the customer does not have to pay for the working output, his or her level of interest in the project success is generally much lower. By this means the original interest of the customer and of the team has to be given as a premise - which did not apply to the NEREID project.

Of course it often happens in real projects, that co-workers are not as motivated as oneself and one has to come to terms with this situation. In the case of the NEREID course with such small teams, it would have been more benefiting for the students to be able to concentrate on the international aspect. This means, for instance, learning which ways of communication and synchronization are the most effectives to fit in an international project or - as another example - how to get on with habits of co-workers of different cultures. If the course is decided to be continued in the next years, it is desirable to check if the different universities attending are taking - not exactly but approximately - the same level of interest in the course. For example, the project leader admitted, that he did not even get a mark for the course, which probably explains his behavior mentioned before regarding the requirements analysis. Nevertheless this could be easily be fixed, if all the participating universities grade the outcome of the course weighted with an equal amount of ECTS European Credit Transfer System points.

5.4.2 Project development

Regarding the project development, the most obvious improvement consists of using a defined process in order to achieve a more structured progress of the project in general. For a small project like the NEREID project, it would have been sufficient to use a list which consists of the single objectives to achieve and work through it step by step. A model like the Rational Unified Process (RUP) Rational Unified Process in such a small team would have meant too much overhead.

Talking about the development, the co-workers in general should release more often and in smaller steps. This shortens time for synchronization and integration of the different modules, because the time one needs for integration is not linear with the lines of code which have to be integrated. That means a doubled amount of code takes more then twice the time to integrate into the project. It is also important to set a rule which ensures that only tested versions are checked in which can be compiled.

Therefore integration tests should be used as well as unit tests. The NEREID team did not make use of automated testing, because it would have caused too much overhead while time was running out.

The time pressure at the end of the project occurred also due to the fact that getting familiar with the technology was underestimated in general. Although the project was quite small and the project leader had experience in the field of ASP.NET, the co-workers had to learn much about the different technologies and particularly their interaction with each other. The students attending the project are now more aware that the advantage of leading an experienced team of developers - who do not have to learn everything from the scratch.

After all, the NEREID project can be counted as a gain in experience for the students and therefore as a success. The problems which occurred showed that the theory lessons about software engineering are true for reality, but also that theory can not replace the practical experience.

Bibliography

[BD09] Bruegge, B.; Dutoit, A. H.:. *Object Oriented Software Engineering Using UML, Patterns, and Java, Third Edition (Pearson International Edition).* Prentice Hall International. 2009.

[Gl98] Glass, R.:. *Software Runaways. Lessons learned from Massive Software Project Failures.* Prentice Hall. 1998.

[NE10] NEREID Wiki:. Website. 2010. http://wwwmatthes.in.tum.de/wikis/nereid/home; visited on May 5th 2010.

[PP08] Paolo Pialorsi, M. R.:. *Datenbankprogrammierung mit Microsoft LINQ.* Microsoft Press Deutschland. 2008.

[Se07] Selby, R. W.:. *Software Engineering: Barry W. Boehm's Lifetime Contributions to Software Development, Management, and Research.* John Wiley and Sons. 2007.

[So10] Software engineering in international teams Wiki:. Website. 2010. http://wwwmatthes.in.tum.de/wikis/sebis/Seminar-Software-engineering-in-international-teams; visited on January 18th 2010.

[Un10] University foreign relations map Wiki:. Website. 2010. http://wwwmatthes.in.tum.de/wikis/nereid/university-foreign-relations-map; visited on January 1st 2010.

CHAPTER 6

Lessons learned

This chapter briefly highlights the key lessons learned of the universities' teaching staff when acting as a host as well as a supervisor regarding the four student projects presented in previous chapters. In doing so, valuable hints and recommendations for future GSE activities in academia are provided substantiated by a succinct explanation.

General course conditions
We strongly recommend that all participating universities agree upon the same student deliverables. In this vein, the project's effectiveness and efficiency can be compared more easily since every student team has to deliver the same result set in the same amount of time. Additionally, standardized software engineering artifacts also facilitate the composition of a common time schedule which can be followed by the teams irrespective the individual project topic. Considering the time schedule, latter should be prepared as early as possible (preferably two months before semester start) allowing students to fit in the NEREID course in their other academic activities. Moreover, we deem a joint kick-off meeting including at least all student team members, the project host, and if possible, all supervising tutors, as being mandatory in order to eliminate initial difficulties endangering the participants' motivation and engagement.

Recommended group size
Assigning 4-6 participants in total to a student team involving 2-3 locations in which each site provides exactly two students has been proven to be successful throughout the four conducted GSE projects. The limited size of the team prompts the team members to equally contribute to the project's success by simultaneously giving the tutors the possibility to personally advice and monitor each individual student. While communication overhead and project's content complexity is reduced for students and supervising tutors, the project sponsor solely needs to keep track of a very limited project scope of 4-6 engineers working for three months on a previously defined outcome.

Project topic
When it comes to the specific project content, we recommend to closely link a project sponsor's focus of research with the proposed topic. Not only the professor in charge would be acquainted with the project's underlying body of knowledge, he or she would also have an increased interest carrying out this project since latter's successful finalization may positively contribute to his or hers chair's research activities. In addition, either the individual content or the type of the project should ensure intensive

communication within each student team. For instance, topics requiring an organizational divide & conquer phase in order to split up and later on integrate the different work packages are suitable since they entail communication among the involved participants. Furthermore, projects with an explicit need for country-specific knowledge and local information generate cross-national exchange as well.

Student target audience
Considering the specific target group, participants should be at least in the 5th semester of their computer science study program, hence late bachelors and master students. On the one hand, those team members are already familiar with elementary software modeling and design techniques (e.g. UML, EPK, Petri nets, and ER models), on the other hand those participants were already in contact with functional and object oriented programming languages almost always required for the fulfillment of the software implementation phase of a project. Being less occupied by the challenging technical but already familiar aspects of a GSE project would allow participants to put more effort into the organizational and communicative part of the course.

Introductory courses
We do not deem extensive preparatory classes taught prior to the actual project start as an indispensable necessity. Tying in with the previous point, participants have been already taught to work on software programs locally and therefore introductory courses would solely convey knowledge regarding project management with regards to global distributed projects. In turn, we warmly recommend to organize a short but crisp GSE overview session in advance in order to make a start more smoothly for all actors by also saving planning and coordination time for the students. This session, which could be easily coupled with an kick-off meeting, should also address escalation paths in the case of team internal and external problems (e.g. definition of project lead, team internal friction, lack of communication with the sponsor), thus whenever the team cannot solve the issue autonomously.

Common evaluation scheme
We suggest to establish a common evaluation scheme for all participating universities making each student's work transparent, comparable, and traceable by generating a more objective picture of the participant's performance at the same time. This scheme, which could be implemented through a simple electronic spreadsheet with the columns representing the three deliverables and rows depicting the universities, is exchanged among all supervising project tutors after the execution phase is finished. Nevertheless, we deem important to keep organizational overhead as low as possible for the tutors. Hence, completing the suggested sheet should not take longer than 10-15 minutes for each project team and its deliverable.

Additional workload
When conducting the different GSE projects, we noticed a higher communication and coordination overhead compared to the overall workload of similar local courses. However, this increased engagement was more than paid off given the insights in research groups of (non-)European countries as well as the international experience we gained ourselves by carrying out those project. We learned how to plan, organize, execute, and coordinate distributed projects by interacting on a cross-country level with different interest groups. We are surprised by the solid results delivered in the short amount of time through the students in hoping to prepare them for upcoming multi-cultural endeavors facing an increasing globalized work environment.

CHAPTER 7

Summary and Outlook

In this document, we presented the outcome of four international student projects which were conducted during the winter term 2009/2010 by means of the final reports. Furthermore, we pointed out the main lessons learned and recommendations from the teaching staffs' as well as the students' point of view. In combination with the approach explained in [Co110], this report may serve as a solid foundation for future GSE activities in academia.

As the NEREID course is considered as being very successful, we will re-offer it in the next winter semester 2010/2011, by taking the experience acquired and lessons learned made into account. Moreover, the group of partner universities will be extended, namely by the Pusan National University in South-Korea as well as the University of Roma located in Italy.

More information on NEREID can be found at our website:

http://wwwmatthes.in.tum.de/wikis/nereid/home

We express our gratitude to all universities which took part in the NEREID course during the winter semester 2009/2010. We are looking forward to jointly organize and host another edition of the course in subsequent winter term.

Bibliography

[Ca99] Carmel, E.:. *Global Software Teams (High Performance Cluster Computing).* Prentice Hall. 1999.

[Co110] :. *Teaching Global Software Engineering and International Project Management - Lessons Learned from Four Academic Projects.* 2010.

[FMS10] Freitag, A.; Matthes, F.; Schulz, C.:. *IT Transformation in the Context of Mergers & Acquisitions - Findings from a Series of Expert Interviews in the German Banking Industry.* 2010.

[Go09] Gotel, O. et al.:. *Quality Indicators on Global Software Development Projects: Does "Getting to Know You" Really Matter?* In *Proceedings of the IEEE International Conference on Global Software Engineering (ICGSE'09).* 2009.

[He07] Herbsleb, J. D.:. *Global Software Engineering: The Future of Socio-technical Coordination.* In *FOSE '07: 2007 Future of Software Engineering.* pages 188–198. Washington, DC, USA. 2007. IEEE Computer Society.

[HH04] Hofstede, G.; Hofstede, G. J.:. *Cultures and Organizations – Software of the Mind: Intercultural Cooperation and Its Importance for Survival.* McGraw-Hill. 2004.

[LH09] Lescher, C.; Hofmann, A.:. *RE'09 Workshop: Collaboration and Intercultural Issues on Requirements: Communication, Understanding and Softskills (CIRCUS).* In *17th IEEE International Requirements Engineering Conference (RE'09).* 2009.

[LHB07] Lescher, C.; Herbsleb, J. D.; Bass, M.:. *Collaboration in Global Software Projects at Siemens: An Experience Report.* In *Proceedings of the IEEE International Conference on Global Software Engineering (ICGSE'07).* 2007.

[LMM08] Lago, P.; Muccini, H.; Muhammad, A. B.:. *Developing a Course on Designing Software in Globally Distributed Teams.* In *Proceedings of the IEEE International Conference on Global Software Engineering (ICGSE'08).* 2008.

[LS09] Laurini, L.; Sol, D.:. *Post Mortem Report, Assessment of the First Experimentation of The NEREID International Project.* page 5. INSA-Lyon, Tec de Monterrey/Campus de Puebla. 2009.

[NP10] NEREID-Page:. Website. 2010. http://wwwmatthes.in.tum.de/wikis/nereid/home; visited on May 5th 2010.

[SMP06] Sangwan, R.; Mullick, N.; Paulish, D. J.:. *Global Software Development Handbook.* Auerbach Publishers. 2006.

List of Figures