Visualizations of Learner Models

Master Thesis

By Vairagkumar Kantilal Godhani
Matr.-Nr.: 316500
vairagkumar.godhani@rwth-aachen.de

Submitted to the Learning Technologies Research Group
LuFG Informatik 9 at RWTH Aachen University

First Reviewer: Prof. Dr.-Ing. Ulrik Schroeder, RWTH Aachen
Second Reviewer: Prof. Dr. Wolfgang Prinz, Fraunhofer FIT

Supervisors: Dr. Mohamed Amine Chatti
Dipl.-Inform. Hendrik Thüs
Christoph Greven, M.Sc. RWTH

31.01.2014

Erklärung | Declaration

Hiermit versichere ich, dass ich diese Masterarbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

I hereby declare that I have created this Master's Thesis completely on my own and used no other sources or tools than the ones listed.

Vairagkumar Godhani
Aachen, 31.01.2014

Abstract

The work of this thesis provides one possible solution to tackle the challenges of lifelong learner modeling such as user interface for open learner modeling, aggregation of learner data, ontologies, reflection and awareness of a learner [2]. As a solution, a JavaServer Faces (JSF) web-application called VisuLeMo (Visualizations of Learner Models) is developed. It allows importing user interests from two learner models named PALM [1] and Webtrace [3] and groups these interests based on the semantic information from Freebase [19]. From the fetched interests' data, visualizations are represented to promote self reflection and awareness of a learner to realize her long term goals.

Acknowledgement

I would like to thank my parents, my family and my friends, for always motivating me in the past to keep on going. Without them this thesis would have never been finished. Prof. Dr. Ulrik Schroeder and Prof. Dr. Wolfgang Prinz for supporting this thesis. Dr. Mohamed Amine Chatti for being a great supervisor, for all his help, his tips and the great times we had during the course of this thesis.

And last but certainly not least Hendrik Thüs and Christoph Greven for always giving me valuable tips, for proofreading this thesis and for the fun we had while working on this thesis.

Table of Contents

Table of Figures

Chapter 1 Introduction

This chapter starts with basic introction by stating the importance of visualization. After that it mentions the lifelong learner modeling [2] and its technical challenges, this is where the motivation of this thesis work comes from. Later section of this chapter explains the problem description and subsequently presents the research question and the aim of this thesis.

1.1 Importance of visualization

Types of visualization

"A picture is worth 1000 words."[1] Visual representation of data gives quick glance and easily understandable format of the content. It also saves time to understand the flow of data. There are different types of visualizations such as Scientific visualization[2], Information visualization[3], Knowledge visualization[4], etc. The Scientific visualization covers visualization of three-dimensional phenomena and is one branch of computer graphics. Our concentration is on Information visualization, which focuses on the use of computer-supported tools to explore large amount of abstract data. In information visualization, different tools and techniques are available. Now which visualization technique to choose, depends on what is the area of interest from abstract data. For an example, when the comparison of number values from data is the area of interest, then it can be represented with bar chart and it can be easily compared and identified which topic has a high or a low value. In contrast to that, when the area of interest is checking increasing or decreasing value with time, in that case a line chart can satisfy the required needs.

Benefits of visualization

Visualization is the representation of an available information in graphical form, so that it can ease the interpretation of data. Large amount of data will be perceived and understood easily by visualizing them. Even simple graphical representation such as pie chart, makes it meaningful to understand the large amount of data. Easily interpreted visualization saves time, cost and energy of a person. These factors are related to self-reflection and awareness of a person. For an example of self-reflection, if an author has published only 2 publications in last year. And if the visualization based on her publications is shown to her then it might reflect

[1] http://en.wikipedia.org/wiki/A_picture_is_worth_a_thousand_words
[2] http://en.wikipedia.org/wiki/Scientific_visualization
[3] http://en.wikipedia.org/wiki/Information_visualization
[4] http://en.wikipedia.org/wiki/Visualization_(computer_graphics)

herself to the counter questions like why I have only 2 publications in the whole year? should I work more and do research to have more publications? For awareness of a person, if visualization based on person's interests is created and shown to her, she will get the overview of it. This can help her to compare herself with the others and she will be aware of it.

1.2 Lifelong learner modeling and its challenges

Learner model and Lifelong learner modeling

A learner model is also known as a student model. A learner can have publications, educational activities, social activities, virtual/web learning activities and activities from learner portals such as L²P[5]. With all these activities, we can create a learner profile, which is represented by a learner model. "A learner model is a model of knowledge, difficulties and misconceptions of an individual. As a student learns the target material, data in the learner model about their understanding is updated to reflect their current beliefs" [4]. Lifelong learner modeling deals with the learner model over a long period of time. "Lifelong learner modeling is the process of creating and modifying a model of a learner who tends to acquire new or modifies his existing knowledge, skills, values or preferences continuously over a longer time span" [1]. An open learner model is part of the lifelong learner modeling. "Open learner models are learner models that can be viewed or accessed in some way by the learner, or by other users (e.g. teachers, peers, parents)" [4].

Roles of lifelong learner modeling

In the research paper [2] Judy Kay and Bob Kummerfeld listed six roles of the lifelong learner modeling and their key technical challenges. Figure 1.1 shows these roles and challenges. One role was to provide open learner modeling (OLM), which had the challenge of user interface that deals with reflection, planning, attention and forgetting. Second role was aggregation of information about the learner from diverse sources, which had the challenge of creating middleware infrastructure for aggregation of data. Another role was interpretation of learner information, which also had the challenge of building user interfaces and new tools for different interpretations. Across the range of roles, one of the recurring technical challenge was user interface of represented data. This refers to an intuitive user interface with the visualization of data. By visualizing information in correct way, user will be able to easily interpret it. For the challenge of aggregation, by importing data from different learner models at one place and visualizing them together will give chance of aggregating information, comparing them and identifying required knowledge.

Recurring technical challenges

[5] http://www2.elearning.rwth-aachen.de/

Roles of a learner model	Technical challenges
1. Open learner modeling (OLM)	User interfaces for reflection, planning, attention and forgetting
2. Aggregation of information about the learner from diverse sources	Middleware infrastructure for aggregation, user interfaces to control aggregation, ontologies
3. Sharing the learner model with people	User interfaces, particularly for privacy management, middleware infrastructure for control of privacy and security, ontologies
4. Interpretation of learner information	User interfaces, new tools for different interpretations
5. Reuse by different applications for the learner's personal use	User interfaces, middleware infrastructure for controlling release of parts of the model and active delivery of parts; ontologies, standarts
6. Institutional use of long term learning data from many learners	User interfaces, middleware infrastructure associated with both sharing and reuse of the model, ontologies, standards

Figure 1.1 : *Roles of the lifelong learner modeling and its challenges [2].*

1.3 Problem description and aim of the thesis

Problem description

Now, we know the importance of visualization from section 1.1. There are already developed frameworks for learner models such as Lifelong Learner Modeling in Academic Networks (PALM) [1] and Lifelong Learner Modeling in Social Networks (Webtrace) [3]. The PALM helps to know the academic interests of a person and the Webtrace helps to know the social interests of a person. The PALM mines information from well known publications using data mining algorithms to allow us to find person's educational interests. The information mined using PALM is very good and efficient, but the data shown is too abstract [1]. In the Webtrace only one visualization is provided and during its evalution it required multiple improvements in the visualization that are mentioned in the future work of [3]. The problem here is that this data lacks good information visualization. The need was to visualize the extracted data from different learner models, in such a way that it will become easier for the user to understand useful data and trends from the information.

Focused challenges and research question

After analyzing roles and technical challenges of learner models as shown in the Figure 1.1 and also keeping in mind the current situation as

mentioned in the last paragraphs, I have decided to focus on below challenges during this thesis work.

- Aggregation of data from different learner models
- Self-reflection
- Awareness
- Usability
- Different visualization techniques

Based on above mentioned challenges, I came to the following **research question**:

"How to best visualize learner models to achieve self-reflection and awareness of the learner ?"

Aim of the thesis

The aim of this thesis is to answer the concept of Judy Kay and Bob Kummerfeld as mentioned in section 1.2 and apply it for different learner models such as the PALM and the Webtrace by visualizing their learner data. Such a framework should support aggregation of information, interactivity, self reflection and awareness of a learner to realize their long term goals. Visualizations of Learner Models (VisuLeMo) should become aggregation point of information for the learner using data from different learner models.

Outline of the thesis

This thesis is outlined into six chapters as described below:

- The second chapter starts with the idea of frameworks for lifelong learner modeling and later describes six different learner models as related work of this thesis.
- Chapter three explains background knowledge of the used technologies in this thesis. Knowledge of these technologies is needed to understand the later presented approach.
- Chapter four illustrates the conceptual approach and requirement analysis of the problem. Afterwards it describes how the solution has been implemented to achieve the goal of this thesis.
- The fifth chapter shows the evaluation of work based on user interviews and improvements made according to user feedbacks.
- The sixth chapter gives a final conclusion of this thesis telling what results are achieved and lists out further improvements which can be done to extend the functionalities of VisuLeMo.

Chapter 2 Related work

As an introduction to this chapter it presents the idea of the frameworks for lifelong learner modeling. Afterwards this chapter describes different learner models which were developed by others and are compared with our approach.

Frameworks for lifelong learner modeling

Long term goals of a learner motivate the lifelong learning process. These goals evolve with the variety of a person's interest fields such as personal, educational and social etc. In the era of web 2.0[6], the lifelong learner models can also be modeled with the electronic media[7]. The usase of electronic media and information technologies in the education field refer to E-learning[8]. Normally, E-learning systems are based on a specific user model which typically represents a combined information of personal data and user interests. A user model refers to a learner model. User modeling is an important research area of Technology Enhanced Learning (TEL) and it addresses the issues of providing and personalizing "the right information at the right time, in the right way" [21]. In the user model, an information related to a specific user is retrieved from different sources and meaningful data is extracted to create a user profile. Framework of the lifelong learner modeling works as a bridge that accomplishes information collection tasks specific to a learner/user and try to represent the information as per learner needs. In this chapter, I will explain six examples of such frameworks.

Checking of learner models

As a part of the related work, I have checked six different learner models: DynMap+ [5], Mr Collins [7], C-POLMILE [8], WILLOW [9], QuizGuide [6], and ViSMod [10]. The learner models are checked based on whether they are providing any aggregation of the data from multiple learner models, whether they are promoting a learner reflection, whether they are providing awareness to a learner, is there any support for different visualization techniques and how is the usability of the system. Figure 2.1 shows answers to all these questions. In the following section of this chapter, I will introduce all these learner models in brief.

[6] http://www.webopedia.com/TERM/W/Web_2_point_0.html
[7] http://en.wikipedia.org/wiki/Electronic_media
[8] http://en.wikipedia.org/wiki/E-learning

Learner models	Aggregation of data from multiple learner models	Promote learner reflection	Provide awareness	Support of different visualization techniques	Usability
DynMap+ [5]	Yes	Yes	No	No	Good
Mr Collins [7]	No	Yes	Yes	No	Poor
C-POLMILE [8]	No	Yes	No	No	Poor
WILLOW [9]	No	No	Yes	Yes	Good
QuizGuide [6]	No	No	Yes	No	Poor
ViSMod [10]	No	Yes	Yes	No	Good

Figure 2.1: *Comparison table of related learner models.*

2.1 Mr Collins

Features of Mr Collins

"COLLaboratively maintained, INSpectable learner model (Mr Collins) aims to encourage greater learner involvement in the construction and repair of the learner model in order to gain a more accurate model, while at the same time promoting learner reflection as a result of this interactive, negotiated learner modeling process"[9]. In the Mr Collins, the information about a learner will be fetched based on the learner inspection. After this, the system and a student can discuss about the information if there is something, which they don't agree. They can also justify what they don't agree through Mr Collins' interface. The Mr Collins was designed to support the conversation among the learner and the system to promote self-reflection [7]. The area of focus in the Mr Collins is object pronouns in Portuguese as a second language learner. It is using only one rule from the twelve rules for Portuguese pronoun placement in the system. An interesting thing in the Mr Collins is that a learner can challenge the system's beliefs if she disagrees with the fact in the system. Then the system will try to justify its measure about the fact to convince the learner. This kind of challenge helps to promote learner reflection [7].

Summary of Mr Collins

In sum, an idea of promoting learner reflection by the discussion of its contents was presented with a small study and an implementation. In the Mr Collins there is no support for the aggregation of the data from the multiple learner models as well as no support for the visualizations. The

[9] http://www.eee.bham.ac.uk/bull/lemore/examples.html

user interface is basic, while an idea of promoting learner reflection is nicely presented and shown by implementing the Mr Collins [7].

2.2 DynMap+

Dynamic visualization of Open Student Models thorough Concept Maps (DynMap+) enables the exploration of individual and group student models [5]. It supports student models from different sources. To understand more about the DynMap+, we first need to understand these terms: An intelligent tutoring system (ITS)[10] and the web-based learning system (WLS)[11]. An ITS is a computer based system which helps a learner for how to use it by providing automatic feedbacks. An ITS provides feedback to the learner using different computer technologies without taking any human help. Main goal of an ITS is to support self learning by using the system. The WLS can be used in the lecture rooms or at home. The WLB helps to create a new learning environment which can include students, instructors, administrators and parents. They provide different access to the involved participants making the WLS available to improve learning [5].

DynMap+ features

An ITS and the WLS collect the data based on student's interaction. The interpretation of this data can be useful for the other students to improve their learning. The DynMap+ visualizes this data based on concept maps[12]. In the visualization, it highlights the important data, which makes it easier to enhance student's learning process. The group student model keeps track of the information for all the steps during learning process. So, the student model knows the evolution in the learning process. The knowledge of a student will change over time; this will lead to update the student model. In sum, the DynMap+ aggregates data from different sources, promote self-reflection and has good user interface [5].

2.3 C-POLMILE

The C-POLMILE is an intelligent learning environment for C programming, which can be used on a pocket PC and a desktop PC [8]. A Pocket PC version of the C-POLMILE enables learning chances when a user does not have a desktop PC. A learner can use either version of the system, which might result in unsynchronized contents. To resolve that an open learner model was required to keep the contents synchronized. With an open learner model, user can update her model when it is necessary. To keep the data synchronized, when a learner model is updated by a user, it will reflect a

[10] http://en.wikipedia.org/wiki/Intelligent_tutoring_system
[11] http://www.pkal.org/documents/WebBasedLearningSystems.cfm
[12] http://en.wikipedia.org/wiki/Concept_map

learner to her previous knowledge. This can help a learner to find misconceptions. It will also assist a learner to clearly understand her learning requirements. Figure 2.2 shows a mobile version of C-POLMILE with a skill meter, illustrating knowledge level, areas of difficulty, misconceptions and the size of the domain.

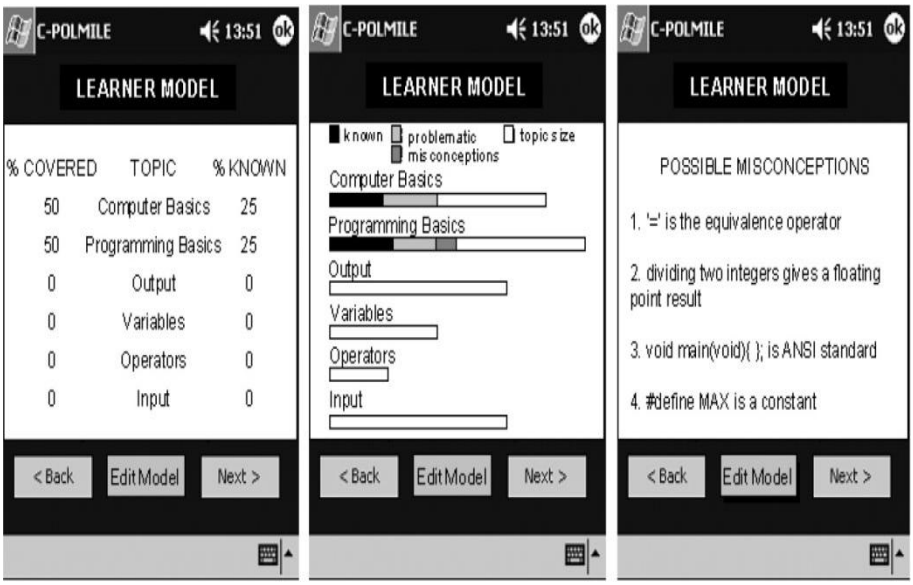


Figure 2.2 : *C-POLMILE's skill meter, illustrating knowledge level, areas of difficulty, misconceptions and size of the domain [8].*

Summary of C-POLMILE

The C-POLMILE presents the skill meter of a learner to differentiate the problematic areas and the misconceptions. It also merges detailed textual description with the graphical view of a skill meter [8]. In sum, the C-POLMILE supports open learner modeling and promotes self-reflection by providing skill meter to find misconceptions. It does not support aggregation of the data from different learner models and does not provide multiple visualizations.

2.4 WILLOW

The WILLOW is a part of the Will Tools, which consist of the different tools such as[13] :

- **WILLOW**: An automatic and adaptive free-text scoring system. A student's conceptual model can be shown in various formats by the WILLOW.
- **WILLED**: An authoring tool
- **WILLOC**: A configuration tool

[13] http://www.eee.bham.ac.uk/bull/lemore/examples.html

- **WILLOV**: A conceptual model viewer. Different student models created by the WILLOW can viewed to the teachers by the WILLOV.

Workflow of WILLOW

The models are automatically generated from free-text answers provided to the WILLOW. In the WILLOW, the students will be asked free-text based questions and answers to these questions from 3 teachers are already in the WILLOW's database. Students' free-text answers are taken and after doing the text operations on them, an ERB algorithm [20] will be applied. The results of an ERB algorithm will give score to the student's answers and also provide text with color schema. For an example, text in green refers to a good point in the student's answer [9].

After that an identification module is applied to find term relevance in the answers. Based on these results, the WILLOW will generate feedback for the students. If a student does not pass the question, she will be asked sub-questions to help her to pass the question. As the WILLOW gets more interaction with the student, it keeps track of how student uses relevant terms extracted from references in the answers. These terms refer to the concepts which have a confidence value. A confidence value says how well the WILLOW believes that a student knows the concept. With this information, a conceptual model of the student will be created and shown to the teacher. This will help a teacher to identify which concepts are misunderstood by student and should be re-evaluated [9]. In sum, the answers to the challenges focused in this thesis for the WILLOW are shown in the Figure 2.1.

2.5 QuizGuide

In the paper [6], Brusilovsky and Sosnovsky mentioned two approaches for encouraging students to use the web-based self-assessment systems to learn programming. The results have shown that these approaches have caused a lot more increase in the system use. This can be helpful when used for educationally beneficial systems. "QuizGuide is an adaptive hypermedia service presenting students with the state of their user model to support navigation through the learning content."[14] Figure 2.3 shows a basic user interface of the QuizGuide. The answers to the challenges focused in this thesis for the QuizGuide are shown in the Figure 2.1.

[14] http://www.eee.bham.ac.uk/bull/lemore/examples.html

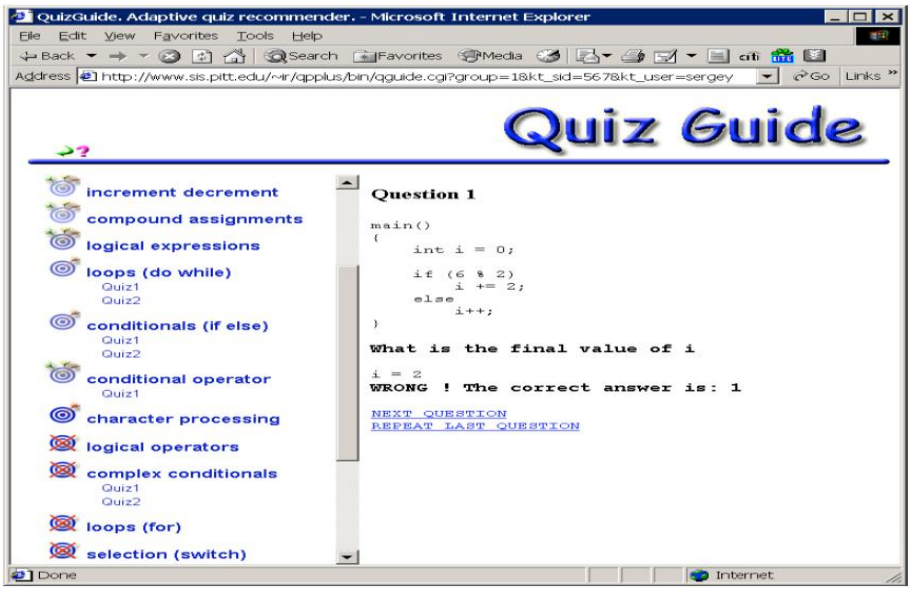


Figure 2.3 : *A screenshot of the QuizGuide user interface [6].*

2.6 ViSMod

Purpose of ViSMod

Visualization of Bayesian Student Models (ViSMod) is based on the Bayesian Belief Networks (BBNs)[15], which are acyclic graph of the random variables connected via their dependencies. A map representing student's social and perceptional information is a Bayesian student model. During the learning process, it also allows annotations from the students and the teachers. An initial version of the ViSMod was an interactive visualization tool (VisNet), with that the students and the teachers can explore the BBNs. The ViSMod is created to support interaction of the students and the teachers with the Bayesian student models[16]. The ViSMod represents student's internal knowledge in front of the teacher and the students and promotes reflection processes with the Bayesian student models in learning [10].

Summary

A Bayesian student model serves as a basis for the discussion between students and teachers by representing the student's own cognitive state. So, the result of the student modeling will be based on the opinions from the system, students and teachers about the characteristics of the student's learning process [10]. In short, the ViSMod helps to promote reflection and awareness by the interaction among users and also supports open learner modeling for the Bayesian student models. Aggregation of the data from

[15] http://en.wikipedia.org/wiki/Bayesian_network
[16] http://www.eee.bham.ac.uk/bull/lemore/examples.html

other learner models and different visualizations are not supported in the ViSMod [10]. A student interface created for supporting the student's interaction with and reflect upon the model is shown in the Figure 2.4.

Figure 2.4 : *ViSMod's student interface supporting interaction [10].*

Summary of the chapter

In sum, the review of related work helped me to understand the past approaches to overcome the challenges of lifelong learner modeling. Keeping in mind this related work, the conceptual approach to achieve the goal of this thesis is presented in next chapter.

Chapter 3 Background

This chapter will give a basic introduction to the topics and techniques used in this thesis. It explains JavaServer Faces (JSF)[17], PrimeFaces[18] and outlines the Freebase API [19]. Moreover, it takes a short look at the Apache Tomcat server [14], Google Gson [15] and Bootstrap [13] in general. After that, the Personal Academic Learner Model API (PALM API) [1] and the Webtrace API [3] are explained in brief. At the end of the chapter, InfoVis [16] and Highcharts [17] libraries which are used for visualization are described. Figure 3.1 shows the MVC architecture of the VisuLeMo for used technologies.

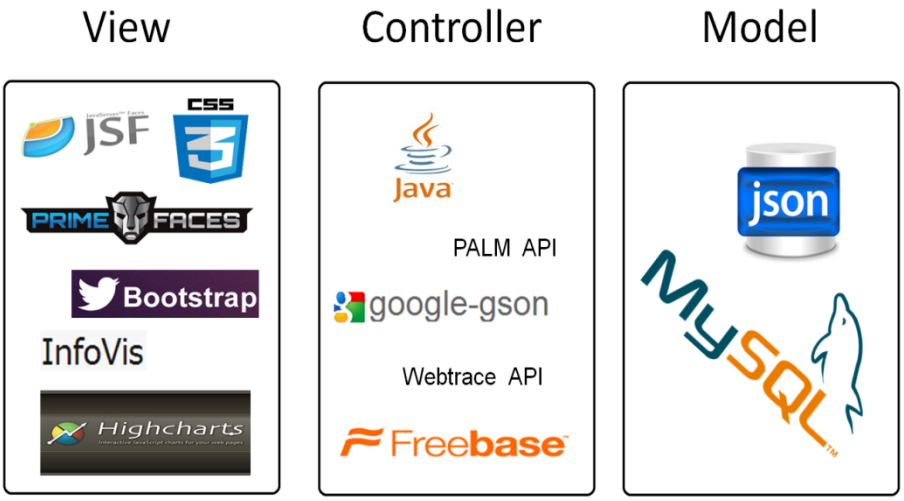


Figure 3.1 : *MVC architecture of the VisuLeMo for used technologies.*

3.1 JavaServer Faces

JSF's separation between presentation and behavior

"JSF technology simplifies building user interfaces for JavaServer applications. Developers of various skill levels can quickly build web applications by: assembling reusable UI components in a page; connecting these components to an application data source; and wiring client-generated events to server-side event handlers"[6]. One main important benefit of using JSF is that it provides clean separation between presentation and behavior of the web application. With JavaServer Pages (JSP), this separation is achieved in part. A JSF application can manage UI elements as stateful objects on the server and map HTTP requests to the

[17] https://javaserverfaces.java.net/
[18] http://primefaces.org/

component specific event handling, while with JSP this is not possible. Because of the separation of logic from presentation, programmers can work independently in different parts of the system and combine them later on. As an example, without knowing programming scripts, a page author can use JSF UI components to link to server-side objects[19].

JSF architecture

JSF has a rich architecture for handling the component state and dealing with the component data. Furthermore, it also has a built-in architecture for handling events and validating user input. As shown in the Figure 3.2, the JSP and JSF technology APIs are layered on top of the JavaServlet API. As a result of this layering, we can have several use cases, for an example using new presentation technology like JSF or PrimeFaces in place of JSP pages. A normal JSF application has one or more backing beans which are connected to the UI components used in JSF pages. An UI component's events and properties are managed by these backing beans, which are ordinary JavaBean classes.

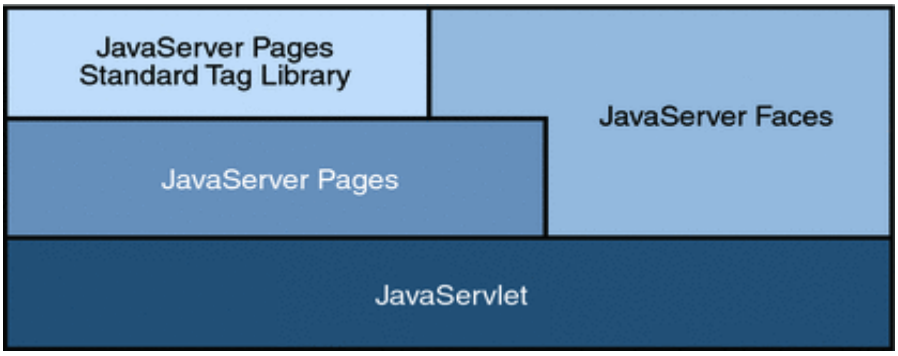


Figure 3.2 : *Java Web Application Technologies*[20].

JSF UI component and Navigation model

JSF technology consists of a UI component model and a Navigation model. JSF UI components are configurable and reusable elements. A component can be simple, such as a button, or compound, such as a table, which can be composed of multiple components. A UI component model includes a validation model, which validates user inputs for editable UI components before updating local values. To specify page navigation and the sequence in which pages should be loaded are handled by JSF navigation model. The lifecycle of a JSF application includes Faces request, Faces response and processing of this request in the intermediate phases, as shown in the

[19] http://docs.oracle.com/javaee/5/tutorial/doc/bnapj.html
[20] http://docs.oracle.com/javaee/5/tutorial/doc/geysj.html#bnadt
[4] http://docs.oracle.com/javaee/5/tutorial/doc/bnaqq.html

Figure 3.3. The intermediate phases support sophisticated UI component model and these phases are[21] :

Phases of JSF application life cycle

- **Restore View Phase**: When the Faces request comes, it starts with this phase and builds the view of the page.
- **Apply Request Values Phase**: Once the component is restored, new value of the component will be calculated with decode() method and then applied to the component during this phase.
- **Process Validations Phase**: In this phase, the validations on the UI components are checked and corresponding events are triggered.

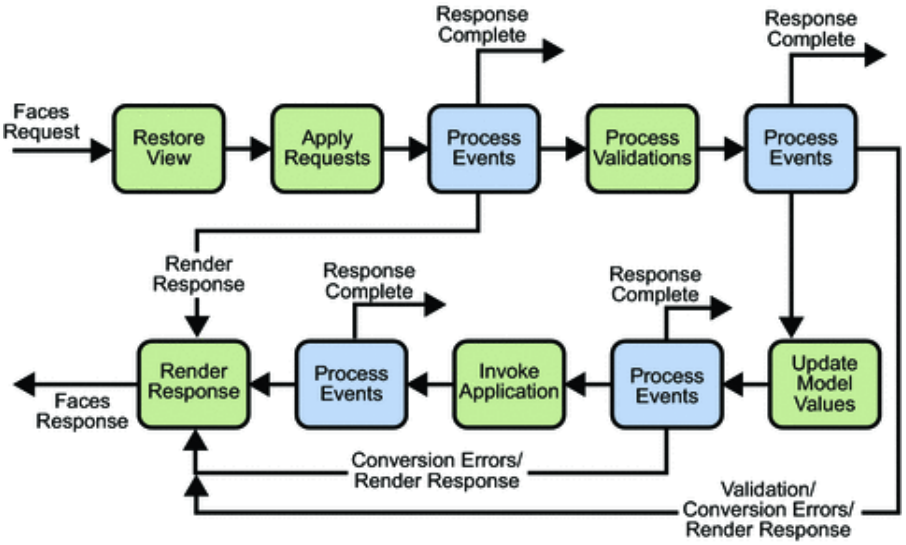


Figure 3.3 : *JSF Standard Request-Response Life Cycle*[22].

- **Update Model Values Phase**: After validating the component values, it will set server-side object values to the component's local values.
- **Invoke Application Phase**: During this phase, application level events are handled.
- **Render Response Phase**: Finally, the components will be rendered as a JSP container in the page.

3.2 PrimeFaces

PrimeFaces[23] is a component library for JSF. It has set of advanced UI components, making the user interface easier to use and simpler compared

[22] http://docs.oracle.com/javaee/5/tutorial/doc/bnaqq.html
[23] http://en.wikipedia.org/wiki/User:Arjant/PrimeFaces

Why PrimeFaces was chosen to use in this thesis?

to basic set of components provided in JSF. One of the technical challenge on which I am focusing during this thesis, refers to the user interface. So, for the better usability of the system, I have decided to use not only JSF but also its component library called PrimeFaces. There are also other JSF component libraries such as RichFaces, ICEfaces, and OpenFaces etc. According to DevRates.com[24], PrimeFaces has become the developers' favorite framework to create rich user interfaces with java with overall rating 9.0 (at the time of writing). Figure 3.4 lists out the important features of the PrimeFaces which help to provide maximum productivity of the software development.

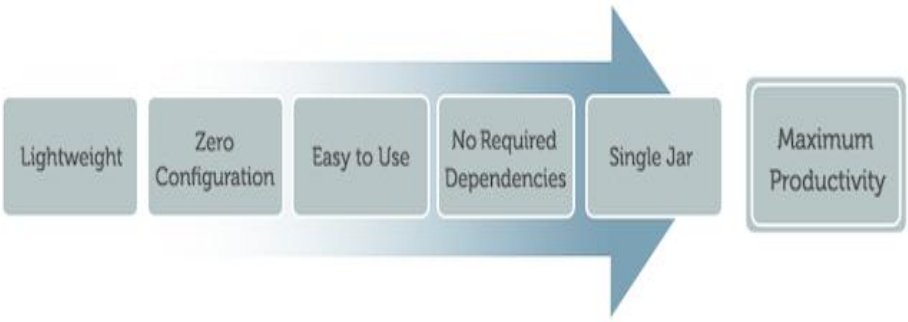


Figure 3.4: *Features of PrimeFaces which lead to maximum productivity.*

Components of PrimeFaces

The earlier version of PrimeFaces was based on the YUI JavaScript library[25]. The current version is a PrimeFaces 3.5, which has over 100 JSF components that are flexible and easy to use with built in Ajax support. These components cover Input fields, buttons, panels, menus, charts, message dialogs, different Ajax widgets, drag/drop, multimedia, file and many more[26]. PrimeFaces has own theme framework, providing 30+ available theme to choose. Built-in PrimeFaces Push framework supports protocols like SSE, JSONP, WebSockets, long polling.

3.3 Freebase API

"Freebase is a large collaborative knowledge base consisting of metadata composed mainly by its community members. It is an online collection of structured data harvested from many sources, including individual 'wiki' contributions. Freebase aims to create a global resource which allows

[24] http://devrates.com
[25] http://yuilibrary.com/
[26] http://www.primefaces.org/

people and machines to access common information more effectively."[27] Freebase API is a collection of HTTP APIs. With the use of these HTTP APIs, it will provide read and write access to the data stored in Freebase. Freebase API has different client libraries for different standards such as for JAVA, for .NET, for PHP etc[28]. A client library consists of JSON and HTTP, so any standard web stack can fetch data and parse the responses from it. In our framework, I have used Google APIs Client Library for Java, as our application is based on JSF. The client libraries help in providing better language integration, making calls that require user authorization simpler and more supporting security[12].

Parts of Freebase API

With the read and write access as mentioned in the previous paragraph, a developer can send query to Freebase for texts, keywords, structured data and images. The Freebase API is further divided into 3 parts[29]:

- **Search API** : Keyword search for entities and other constraints. The response of the search services consists of the Freebase data for a given entity and also from other sources like Wikipedia content.
- **Topic API**: All the information for an entity will be given as a summary. The Topic API is a web service. The response from it contains all the known facts for queried topic with the images and texts.
- **MQL Read/Write APIs** : For an entity or collection of entities, structured data can be fetched and written by these APIs. With these APIs, a developer can perform structured queries on the Freebase. It uses Metaweb Query Language. This enables easy use of the graph data from the Freebase. For an example, questions like "*singers of America music album*" or "*rivers in Germany*." can be answered by queries.

Freebase Suggest Widget and filtering

The Freebase API also provides 'Freebase Suggest Widget', which is a built in JQuery widget for picking entities. This widget can be placed on the website as a plug-in to embed Freebase search on the site. Figure 3.5 shows a 'Freebase Suggest Widget' on the Freebase site with the suggest results of a topic 'software'. Freebase's search API supports variety of filtering constraints[30], to get the correct required information from the query and removing unwanted results. There is a query parameter called 'filter', in

[27] http://en.wikipedia.org/wiki/Freebase
[28] https://developers.google.com/freebase/v1/libraries
[29] https://developers.google.com/freebase/v1/getting-started
[30] https://developers.google.com/freebase/v1/search-output

which we can specify filter constraints. For an example, using 'type' filter constraint with the value '/people/person'. This will limit the response of the query to only domain with a value 'people' and type of the domain with a value 'person'. Shown below is an example of the mentioned filter constraint.

```
filter=(any type:/people/person)
```

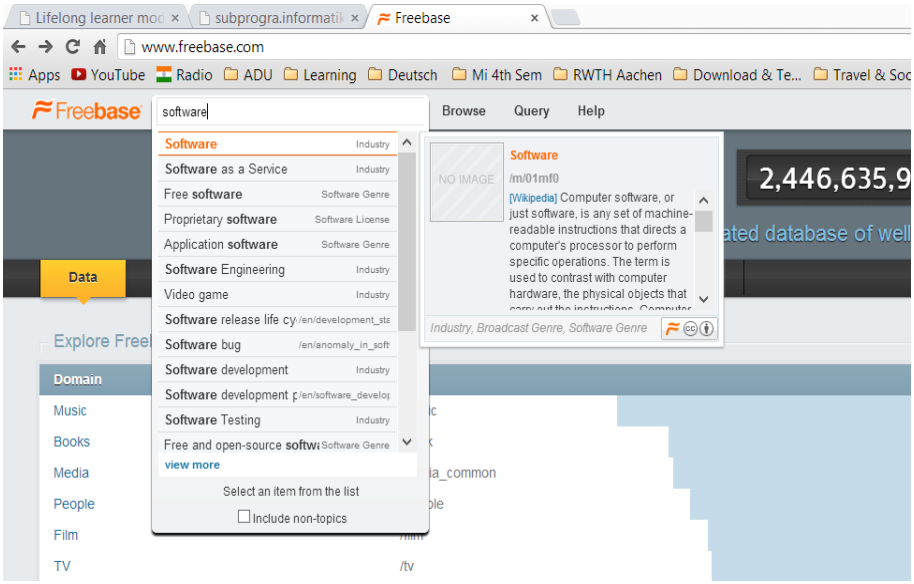


Figure 3.5: *'Freebase Suggest Widget' with the suggest results of a topic 'software'*[31].

Logical operators for filtering

It is also possible to combine more than one filter constraints and use logical operators between them to clearly specify which results are requested as the response. Freebase filter constraint supports 4 filtering operators, which are as below:

- **any**: Works as logical OR operator.
- **all**: Works as logical AND operator.
- **not:** Works to negate the option.
- **should**: Specifies that the constraint is optional.

As an example, to get the results with a type '/people/person' and with a domain '/film', filter constraint will look like this:

[31] http://www.freebase.com/

```
filter=(all type:/people/person domain:/film)
```

Example of Freebase API query

Common URL convention for the Freebase API query is

```
https://www.googleapis.com/freebase/v1/search?query=research
```

where, research is a queried topic for which the query is asked to get the Freebase data. There are also other query parameters that can be used to limit and filter the response data. In this thesis, I am using Freebase API for clustering learner interests by their categories and parent categories. In the Freebase request to get interest's categories, I am using a parameter 'output=(type)', because I am interested in fetching different types under which particular interest comes. I am also using a second parameter 'limit=1' to fetch the first topic for the requested query which has highest score. The sample URL with these 2 parameters will look like this:

```
https://www.googleapis.com/freebase/v1/search?query=research&output=(type)&limit=1
```

Freebase terminology

In Freebase, all the search terms are mainly divided into below 3 parts:

- **Topics:** Freebase documentation says that they have over 39 million[32] topics related to real-world entities such as places, things, and people. The data in the Freebase is represented by graphs and nodes in these graphs correspond to the topics. Some topics are important as they hold a lot of data. While other topics are important because they link to other topics, mainly in the different domains. For an example, some topic such as love, emotion etc. do not have more properties but on the other hand they are being used very often in movie subjects, music albums, books etc. which makes them important[15]. In this thesis, I am mapping learner's interest to topic in the Freebase. See the Figure 3.6.
- **Types:** One topic can come under several categories. For an example, Michael Jackson was a musician, singer, songwriter, arranger, dancer, entertainer, choreographer, actor, businessman[33].To represent multi-faceted nature of the topics like

[32] https://developers.google.com/freebase/guide/basic_concepts#topics
[33] http://www.michaeljackson.com

Michael Jackson, Freebase uses *types.* In short, *type* is a collection of the Freebase topics. Topics of Freebase can be assigned to any number of types in which they belongs. For the example above, topic about Michael Jackson will be in many types like musician type, singer type, dancer type etc. We can think of types as a conceptual container of the topics, which share similar information or properties. In this thesis, I am mapping the types of Freebase with the categories of interest as shown in the Figure 3.6. By this way, grouping of learner interests can be done. For an example, learner's interests like Football and Lionel Messi will come under the category/type called Sport.

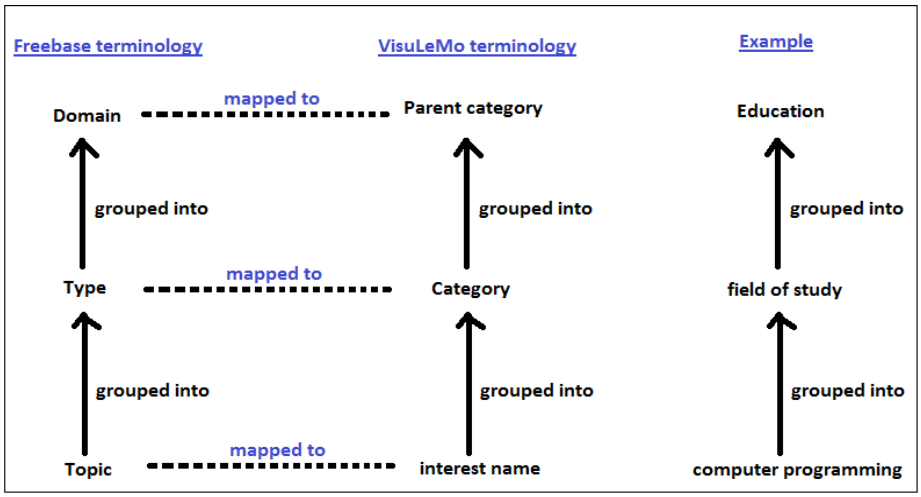


Figure 3.6: *Mapping of Freebase to the VisuLeMo terminology with an example.*

- **Domains**: The way, topics in the Freebase are grouped into types, the types themselves are grouped into domains. The domains are more generalized topics. For an example, Business, Entertainment, Life Style, Education etc. are domains and types like Sport, Music, and Movies can come under the domain called Entertainment. One domain can have any number of types, which share similar information or properties. In this thesis, I am mapping a domain of Freebase with the parent category of the interests as shown in the Figure 3.6. A parent category/domain is a group of categories/types. And the category/type is a group of interests/topics.

Summary of Freebase

In sum, Freebase is owned by Google[34] and it powers the new Google Semantic search[35]. It is a graph database with more than 39 million entities that will be used to refine the Google search results in the future. Its API and the interface have been proved as evolved. To take all these advantages and to check how the results based on Freebase API clustering for the learners' interests will be beneficial, I have chosen the Freebase API. There are other semantic knowledge bases compared to Freebase such as Wikipedia[36], Yet Another Great Ontology (YAGO)[24] etc. In the next sub-section of this chapter, the Wikipedia semantic and the YAGO are explained in brief.

3.3.1 Wikipedia semantic

Alternative 1 of Freebase : Wikipedia

Wikipedia is the free encyclopedia that everyone can edit. It is hosted by the Wikimedia Foundation[21], which hosts so many other projects such as MediaWiki, Meta-Wiki, Wikinews, Wikisource, and Wikibooks etc. The Wikipedia knowledge base is being expanded continuously by the user collaboration. The users of the Wikipedia can add new articles or update the existing articles. The MediaWiki[37] is a web service API, which enables access to wiki data, meta-data and wiki features. The wiki features can be pictures, articles and the refined contributed results of the active users from the Wikipedia.

3.3.2 Yet Another Great Ontology

Alternative 2 of Freebase : YAGO

Yet Another Great Ontology (YAGO) is one of the sub-projects of the YAGO-NAGA project[38] and it was developed at the Max-Planck-Institut Informatik in Saarbrücken, Germany. It is a knowledge base which is extracted from the other sources such as Wikipedia. In its own home page it is defined as "YAGO is a huge semantic knowledge base, derived from Wikipedia, WordNet and GeoNames. Currently, YAGO has knowledge of more than 10 million entities (like persons, organizations, cities, etc.) and contains more than 120 million facts about these entities."[39] In YAGO, the relations between the entities are given their confidence value. Its accuracy has been proved and manually evaluated with value of 95%[21]. YAGO is anchored based on time and space. It merges the Wikipedia category system with the

[34] https://www.google.com/
[35] http://davidamerland.com/google-semantic-search.html
[36] http://en.wikipedia.org/
[37] http://www.mediawiki.org/wiki/MediaWiki
[38] http://www.mpi-inf.mpg.de/yago-naga/
[39] http://www.mpi-inf.mpg.de/yago-naga/yago/index.html

taxonomy of WordNet[40] in various classes. Moreover to that, YAGO ontology also adds WordNet domains like "art" or "history".

3.4 PALM API

PALM basic

PALM mines the information from the well known publications using data mining algorithms to allow us to find person's academic interests. The information mined using PALM are very good and efficient, but the data shown are too abstract [1]. The whole work flow of PALM was divided into 3 main modules: 1) data retrieval, 2) interest mining, and 3) visualization modules as shown in the Figure 3.7. According to [1], PALM was designed for enabling access to large number of online publications related to academic learning. In PALM, academic interests of a learner are collected, mined and then aggregated to create an academic learner model. And finally, these data are being visualized in an abstract way.

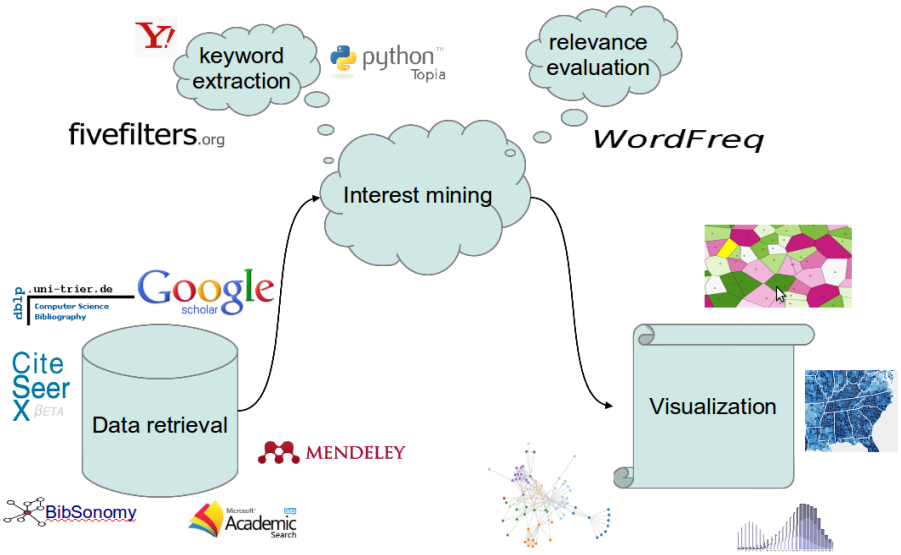


Figure 3.7 : *Conceptual approach of PALM showing its 3 main modules and how they are related [1].*

Why PALM was chosen to use in this thesis?

For everyone, who wants to enhance the functionalities of PALM, it provides PALM API at [11]. This API accepts a standard HTTP GET method and returns the results, which are rendered in the JSON format. As mentioned in the previous paragraph, the visualization in PALM was to abstract and basic. So it was decided to visualize it in a better way and also combing them with the interests from the other learner models. To do that I am using the PALM API for the interests. Figure 3.8 shows a part of the

[40] http://wordnet.princeton.edu/

PALM API JSON response for the interests of an example author. The same result with full JSON response can be viewed at [12].

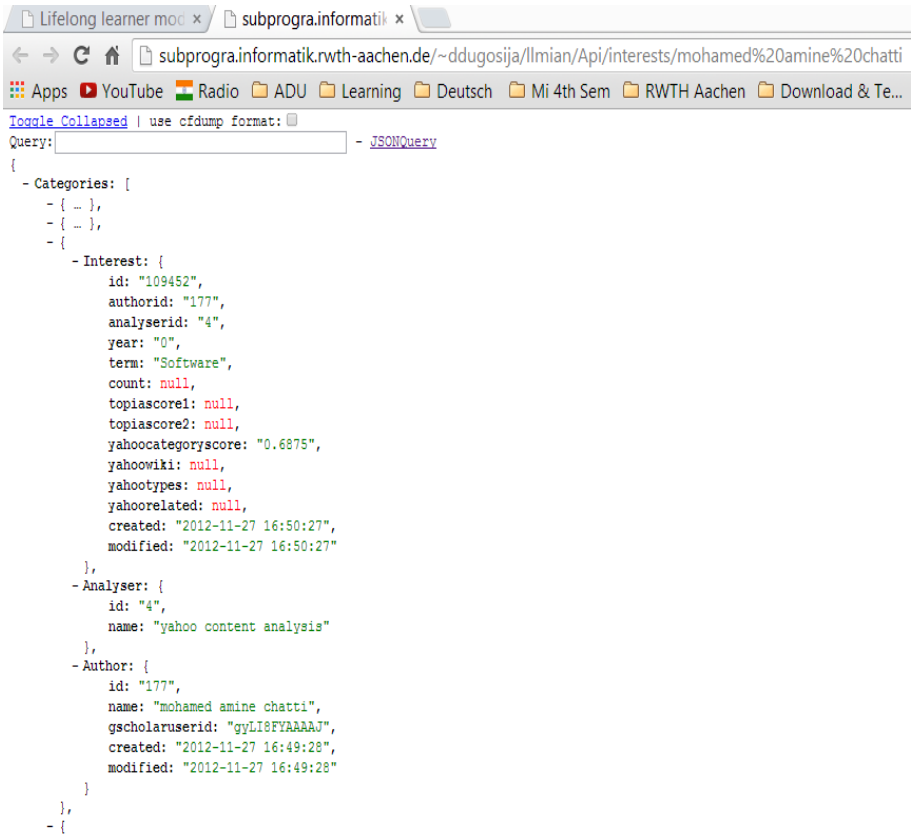


Figure 3.8 : *Part of the PALM API JSON response for the interests of an example author.*

3.5 Webtrace API

Webtrace basic

The Webtrace was the name given to the web application developed in the thesis 'Lifelong Learner Modeling in Social Networks'[3]. It collects information from the web services and the social networks to create a learner profile, which depicts mainly the social interests of a learner. It uses the Wikipedia semantic[41] to categorize these interests into groups. At last, a learner profile is visualized based on the interests and a learner can explore it interactively. In the thesis work, they provided an algorithm which solves the mentioned task with a good quality of a learner profile [3]. Currently, the running Webtrace application can be viewed at [18].

[41] http://semantic-mediawiki.org/

In the VisuLeMo, I wanted to use the social interests of a learner collected in the Webtrace. To do that, I need to request these interests from the Webtrace API. At the moment, the Webtrace doesn't have any API which provides access to the social interests from outside. However, the thesis work of the Webtrace was done at Learning Technologies Research Group, RWTH Aachen University, where the work of the VisuLeMo is also done. So, I get the access to the database of the Webtrace to use it in the VisuLeMo for the social interests of a learner gathered in the Webtrace.

Why Webtrace was chosen to use in this thesis?

3.6 Bootstrap

Bootstrap[42] is a front end framework used in the web applications. It has HTML and CSS based templates for various UI components and also has some JavaScript extensions. Bootstrap was developed at Twitter[43] and later it was released as an open source[44]. Using Bootstrap in the project is very easy. Just download the Bootstrap library, put the CSS and JS files in the project folder and at last include these files in the view pages and use it. Bootstrap has a built-in support for responsive pages, which means it automatically adapts to the different screen sizes. It has a special support for responsiveness of the images, tables and the navigation menus. Bootstrap contains a mobile first and responsive fluid grid system that has 12 columns as shown in Figure 3.9, which scales according to the device screen size.

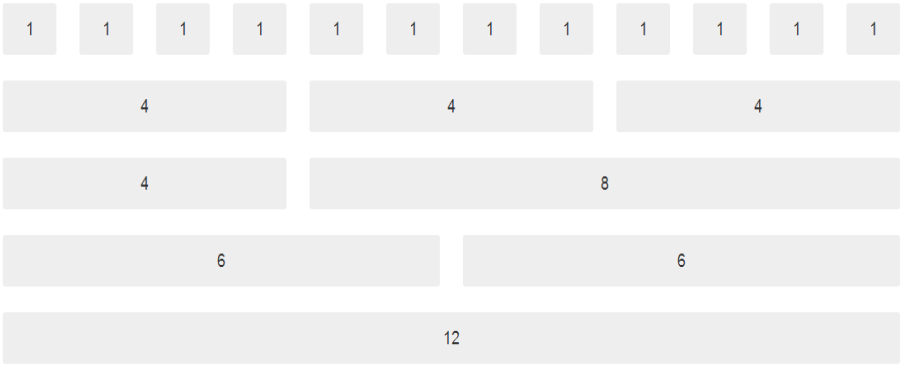


Figure 3.9 : *Bootstrap's fluid grid system with 12 columns.*

It is decided to make the VisuLeMo framework responsive, so that it will work with different screen sizes enabling its usage in various mobile devices and tablets. Bootstrap is designed such that it supports all the latest

Why Bootstrap was chosen to use in this thesis?

[42] http://getbootstrap.com/
[43] https://twitter.com/
[44] http://en.wikipedia.org/wiki/Bootstrap_(front-end_framework)

browsers on desktop and mobile devices, it might display slightly different on the older browsers [13].

3.7 Apache Tomcat

An Apache Tomcat is an open source web server, which is built in java and also referred as a Servlet/JSP container. It provides environment to run java code. An Apache Tomcat has configuration and management tools for specifying the server parameters. These parameters can also be configured in the XML configuration files [14]. In this thesis, I am using an Apache Tomcat server version 7.0.34.0. An Apache Tomcat was released with 3 main components[45] as described below:

- A Servlet container called Catalina
- An HTTP connector called Coyote
- A JSP engine called Jasper

3.8 Google Gson

Gson basic

Gson is a Java library from Google[46] which can parse the JSON string to the Java objects and can also convert the Java objects to the JSON string. It can still parse the JSON string for the pre-existing Java objects that you do not have source code [15]. It works on the concept of serializing and de-serializing the Java objects to and from the JSON strings. It has very simple methods called toJson() and fromJson() to parse Java objects to JSON and vice-versa. Gson can work with nested classes, collections and generic types[47]. When de-serializing, it removes extra fields present in the JSON input string. This handles null object fields from the output. One useful feature for the developers is compact and pretty printing of the objects, which becomes helpful while debugging the code[24].

Why Gson was chosen to use in this thesis?

In this thesis, the framework is reading learner's academic interests from the PALM API. The response of these API is in JSON string and it is using Gson to parse this JSON string to the Java objects. It is also sending request to the Freebase API, to get interest's categories and parent categories. The response of a Freebase request is also in JSON string and it is using Gson to parse it to the Java objects and then use it in our code.

[45] http://en.wikipedia.org/wiki/Apache_Tomcat
[46] https://www.google.de/
[47] http://en.wikipedia.org/wiki/Gson

3.9 JavaScript InfoVis Toolkit

InfoVis features and Purpose for this thesis

The JavaScript InfoVis Toolkit[48] is a JS library to create interactive data visualizations for the Web. It provides different kinds of visualizations covering the basic visualizations like Area charts, Bar charts, Pie Charts and also the advanced visualizations such as Sunburst, Icicle, ForceDirected, SpaceTree, TreeMap, HyperTee, RGraph etc. These advanced visualizations are useful specially to visualize complex, hierarchical or structured data [16]. Figure 3.10 shows an example of Icicle visualization using InfoVis Toolkit. For a visualization using InfoVis, first we need to create a JSON object of the data representing the structure of visualization. Then this JSON object is passed to the init() method of that specific visualization to see the graph. Utility functions like merge(), getparent(), getsubtree() etc. are provided to do graph operations. It also provides visualization options, graph plotting, graph manipulation and graph labeling options [16]. And finally the graph will be rendered on the canvas according to the options set for a visualization. The InfoVis Toolkit was chosen for the advanced visualizations such as SpaceTree, HyperTree and Icicle in the VisuLeMo.

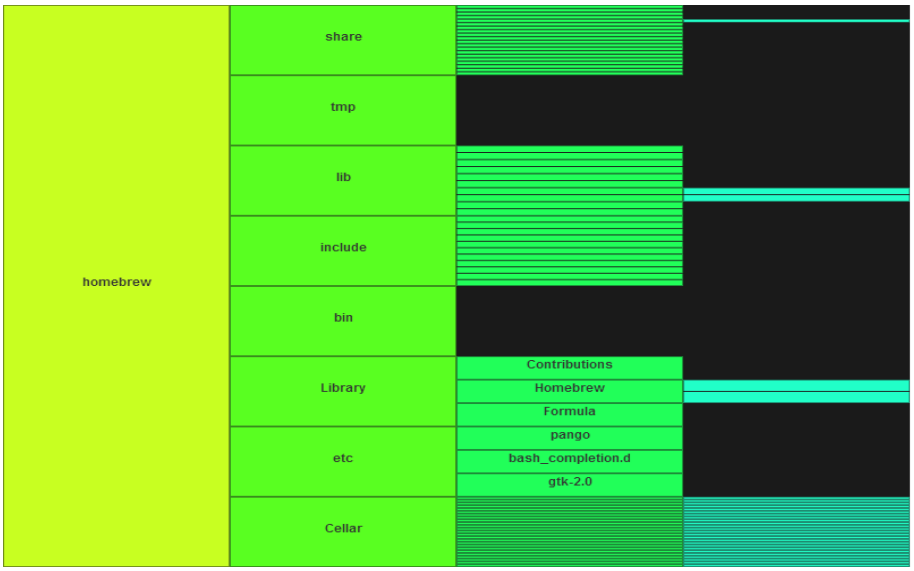


Figure 3.10: *An example of Icicle visualization using InfoVis Toolkit[49].*

3.10 Highcharts JS

Highcharts JS[50] is a JavaScript and HTML5 based chart library providing features to create web based interactive charts. Highcharts offers different

[48] http://philogb.github.io/jit/
[49] http://philogb.github.io/jit/static/v20/Jit/Examples/Icicle/example2.html
[50] http://www.highcharts.com/

Highcharts JS features and Purpose for this thesis

graph types such as line, spline, area, areaspline, column, bar, pie, scatter, angular gauges, arearange, areasplinerange, columnrange, bubble, box plot, error bars, funnel, waterfall and polar chart types[51]. Figure 3.11 shows an example of a line chart explaining the different parts of a chart in Highcharts. Highcharts JS provides very good feature for filtering contents within the charts and because of these easily usable filters on the contents within the charts, I have chosen to use this library for basic visualizations of learner interests.

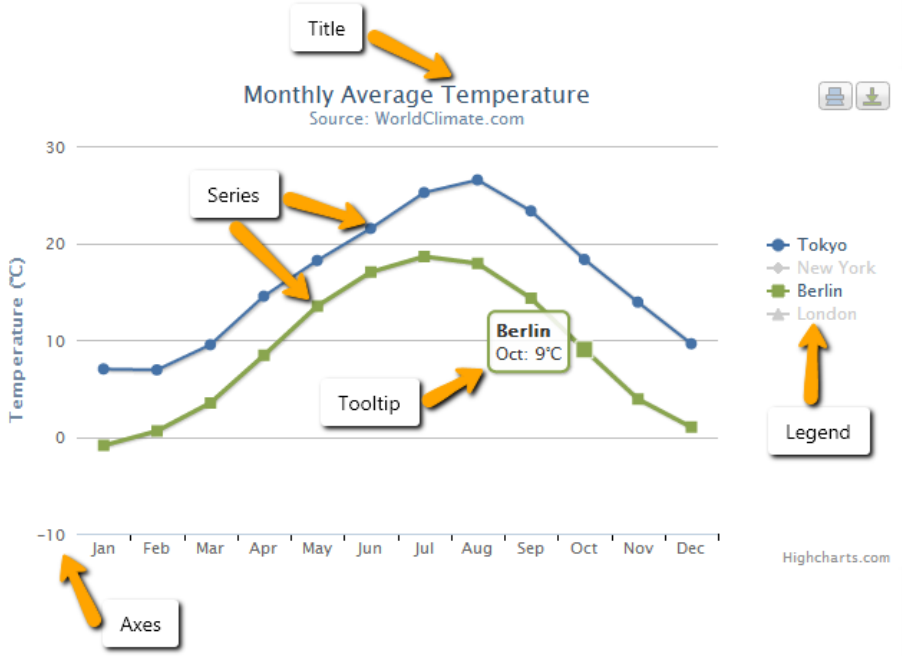


Figure 3.11 : *An example of a line chart explaining the different parts of a chart[52].*

Two parts of Highcharts JS

Highcharts JS is mainly divided into 2 parts. First part is a Highcharts, offering all the chart types that are mentioned in the first paragraph. It also provides very good demonstration of the different chart types with the possible options to customize charts at [28]. To create combination chart by combining the different chart types in one graph is very easy in Highcharts. As shown in the Figure 3.12, we just need to create series of the different chart types in one graph. Highcharts has a facility to export the rendered chart in PNG image, JPEG image, PDF document or SVG vector image [17]. Second part is a Highstock[53] with which we can create stock or timeline

[51] http://www.highcharts.com/demo/
[52] http://www.highcharts.com/docs/chart-concepts/understanding-highcharts
[53] http://www.highcharts.com/products/highstock

charts. Highstock specially offers navigation options within the graph such as a data range selection, navigator series, scrolling, date picker and panning. Figure 3.13 shows an example of Highstock chart with two panes with the navigation options: date picker, data range and panning. Full demo of all the available charts from Highstock is available at [54].

```
series: [{
    type: 'column',
    name: 'Jane',
    data: [3, 2, 1, 3, 4]
}, {
    type: 'column',
    name: 'John',
    data: [2, 3, 5, 7, 6]
}, {
    type: 'column',
    name: 'Joe',
    data: [4, 3, 3, 9, 0]
}, {
    type: 'spline',
    name: 'Average',
    data: [3, 2.67, 3, 6.33, 3.33]
}]
```

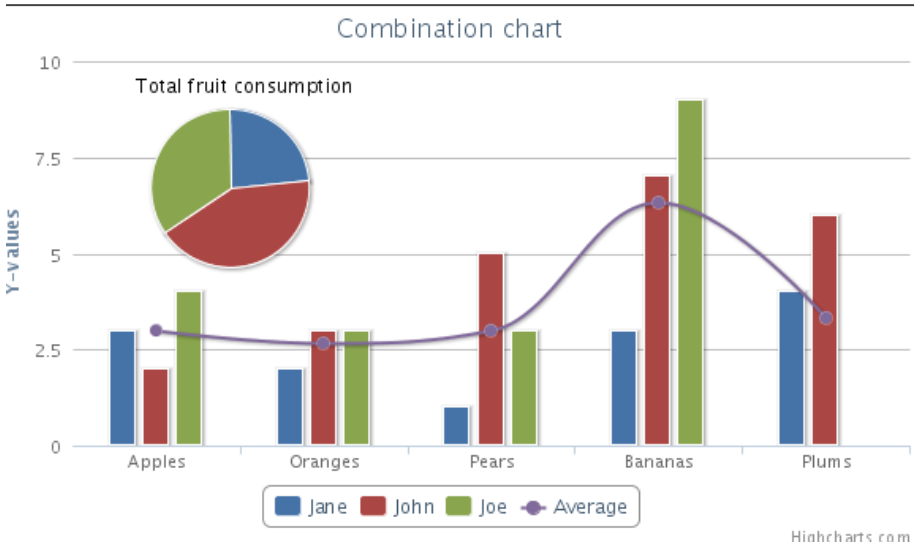


Figure 3.12 : *An example of a combination chart showing series of different chart types in one graph*[55].

[54] http://www.highcharts.com/stock/demo/
[55] http://www.highcharts.com/docs/chart-and-series-types/combining-chart-types

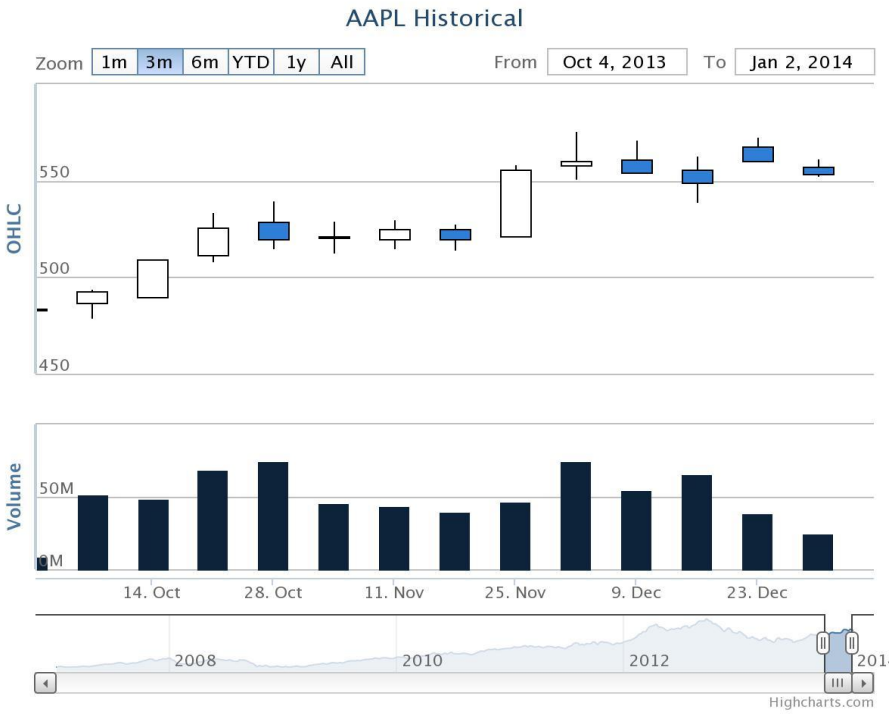


Figure 3.13 : *An example of Highstock chart with two panes with the navigation options: date picker, data range and panning.*

Summary of the chapter

In sum, the background information about the technologies and APIs that have been explained in this chapter will help to understand the implemented solution of this thesis in the next chapter.

Chapter 4 Conceptual Approach and Implementation

This chapter illustrates conceptual approach to solve the problem of this thesis and then lists out the requirements of solution framework. Afterwards, it describes how the solution has been implemented to achieve the goal of this thesis.

4.1 Conceptual approach

The conceptual approach of this thesis is divided into four main objectives: Data retrieval, Mapping to an internal data structure, Clustering the data and Visualizations. Figure 4.1 represents conceptual approach for solving the problem of this thesis.

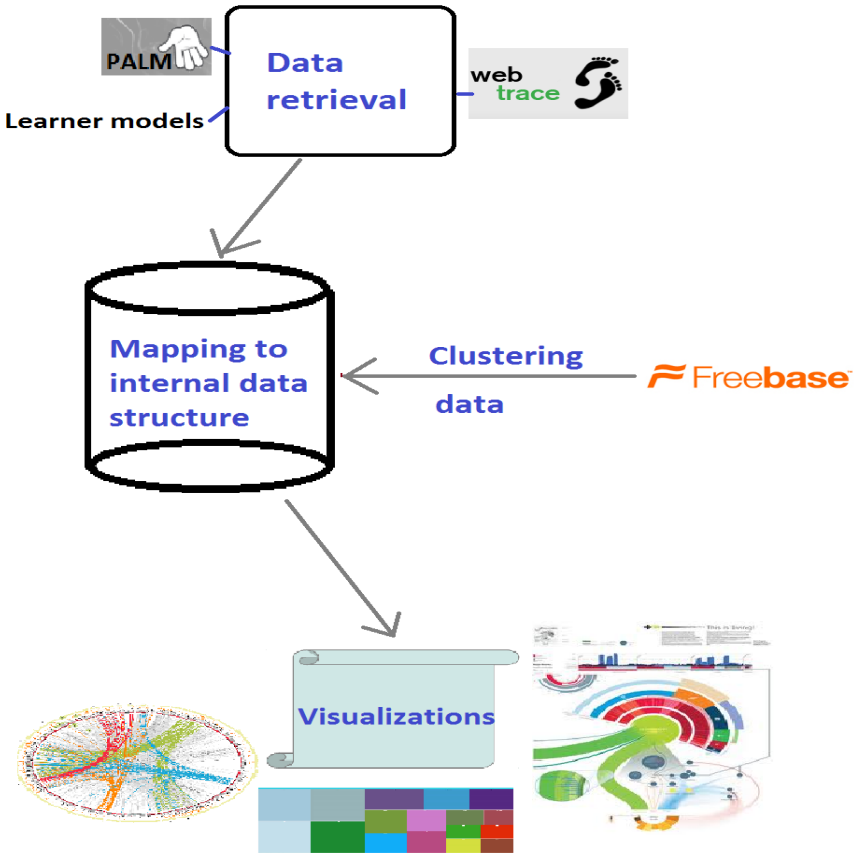


Figure 4.1: *Conceptual approach of the VisuLeMo.*

4.1.1 Requirement analysis

Here, requirement analysis is list of features which are required to have in the solution to achieve the goal of this thesis. Keeping in mind the aim of this thesis as mentioned in chapter 1 and the objectives from the conceptual approach of Figure 4.1, the solution should be able to:

Requirements

- provide user authentication and management of the personal details and settings for learner model Ids.
- have user interface with good usability.
- support importing learner data from different learner models.
- have an internal data structure which can be mapped to the fetched data from learner models.
- group the fetched interests using one semantic knowledge base (i.e Freebase).
- support open learner modeling by allowing a user to edit her interests.
- provide multiple visualizations to visualize the results. The visualizations should be interactive and support filtering to promote self-reflection and awareness of a user.

To understand clearly how above requirements can be realized, I have created the mockup of VisuLeMo using Basalmiq[56]. Appendix B shows screenshots of this mockup. It includes the basic GUI of VisuLeMo, an idea of importing data from learner models and visualization techniques. This mockup is checked considering the Use cases mentioned in section 4.1.2 to clearly define the work flow of the application.

4.1.2 Use cases

Definition and purpose of Use case

"A use case is a methodology used in system analysis to identify, clarify, and organize system requirements. The use case is made up of a set of possible sequences of interactions between systems and users in a particular environment and related to a particular goal."[57] Use cases represent scenarios of users interacting with the system to perform specific task. Use cases also cover possible failure and success cases while performing the task. They are made in early phase of system design and describe more general functionalities of the system. Use case contains name of the use case, work flow of the task, involved users and other use cases if any [26]. Use cases are used to understand and evaluate the work flow of system.

[56] http://balsamiq.com/
[57] http://searchsoftwarequality.techtarget.com/definition/use-case

Below, three use cases are explained which cover main work flow of the VisuLeMo application.

User profile creation and updation: A user visits the VisuLeMo web page and reads 'About VisuLeMo' page, which motivates her to use VisuLeMo. To use VisuLeMo, a user needs to have login credenticals which cover two scenarios. First, a user goes to create new user page, fill in the form and successfully gets the credentials to access VisuLeMo functionalities. Second, a user has created a profile in the past and interested in updating her profile. She authenticates herself with the previously created username and password and goes to the 'settings' page. In the 'settings' page, a user sees her details and is able to update her personal details and Ids for different learner models.

Collecting user interests: A user have already created account in VisuLeMo and wants collect her interests from different learner models. A user clicks on 'Fetch interests' link, selects learner models from which she wants to collect her interests. In the next page, a user will see all her collected interests. At the time of collecting interests, these interests' categories are also imported from the Freebase. Later from the 'My interests' tab, a user is able to edit the collected interests and it is also possible to view categories of each interest.

Visualizing interests: A user wants to get impotant information from her interests by visualizing them. A user has already collected her interests from different learner models. A user goes to 'Visualizations' page and sees different names of visualizations. The names of visualizations tell a user about what information she will get from the visualization. A user clicks on the visualization based on her interested information and sees graphical representation of interests. A user interacts with the graph to filter the contents. There is also visualization for hierarchy of interest categories. A user can browse through layers of categories from general to specific interest.

4.2 Implementation

Four parts of the implementation

After gathering the requirements of services that are needed to achieve the solution for this thesis, I have divided the implementation of solution into four parts:

- A Graphical User Interface (GUI) of the VisuLeMo.

- Importing and parsing data from different learner models to the internal data structure of VisuLeMo.
- Clustering the fetched interests based on Freebase.
- Choosing appropriate visualization techniques and visualizing learner interests.

In the following sections, each of above mentioned four parts is explained in details.

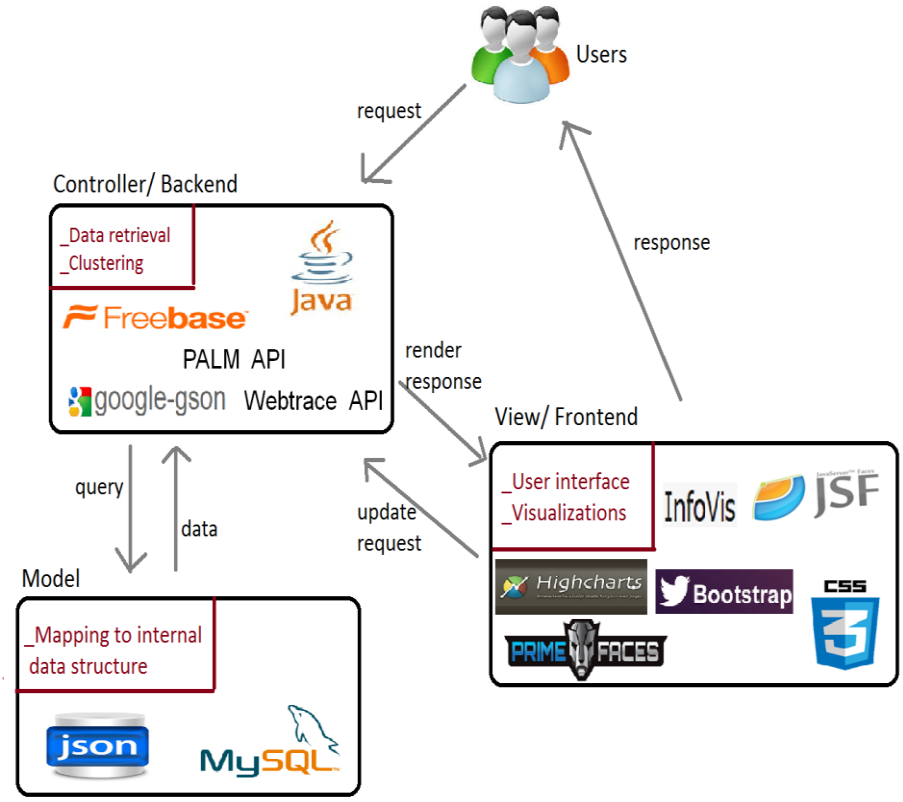


Figure 4.2 : *Technical architecture showing communication between front-end and back-end of VisuLeMo application.*

Functionalities of VisuLeMo interface

First part of an implementation was the GUI of VisuLeMo. The GUI created for VisuLeMo supports various functionalities covering basic options such as registering a new user, send an email on forget password, user authentication, viewing and updating user settings, viewing and updating personal interests. It also covers advance options such as importing data from different learner models and visualizing them with the various visualizations supporting user interaction. The GUI of VisuLeMo is responsive, which means it adapts to the change in screen sizes of browser. This enables VisuLeMo's usage in different mobile devices and tablets. An

initial, HTML-CSS template was taken from [23]. Afterwards, for better structuring of view pages, each XHTML[58] page of the JSF application was divided into 4 separate templates as shown in the Figure 4.3. These 4 templates were:

Four templates of view page

- **head.xhtml:** It contains the part of an XHTML page that should come in Head tag like including CSS files or JS files etc.
- **navigation_header.xhtml**: It contains navigation part of the application.
- **footer.xhtml:** It contains footer part of all the pages.
- **Page's own XHTML file:** It includes 3 templates listed above at appropriate place and body content specific to this page (see Figure 4.4).

A final version of GUI is the outcome of sub-sequent phases of GUI, which have evolved during thesis. These sub-sequent phases include mockup of the framework made in Balsamiq[59], initial JSF application with the basic functionalities and improved GUI with CSS templates and Bootstrap. See Appendix B and Appendix C for the screenshots of VisuLeMo GUI.

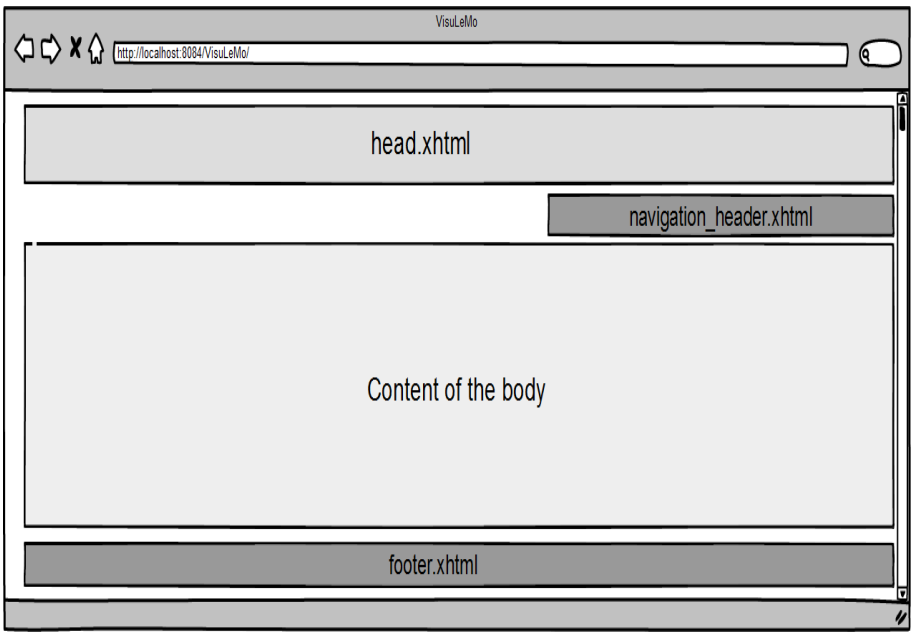


Figure 4.3 : *One XHTML view page is divided into 4 separate templates.*

[58] http://xhtml.com/en/xhtml/reference/
[59] http://balsamiq.com/

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://java.sun.com/jsf/html"
      xmlns:f="http://java.sun.com/jsf/core"
      xmlns:ui="http://java.sun.com/jsf/facelets"
      xmlns:p="http://primefaces.org/ui">
    <h:head>
        <ui:include src="head.xhtml" >
            <ui:param name="title" value="VisuLeMo | Login"/>
        </ui:include>
    </h:head>
<body>
<div class="spinner"></div>
<!-- header -->
<header>
    <ui:include src="navigation_header.xhtml" >
            <ui:param name="active_menu" value="Login"/>
    </ui:include>
</header>
<div class="bg-content">
    <div class="container">
        <div class="row">

            Content specific to this page!

        </div>
    </div>
  </div>
<!-- footer -->
<footer>
      <ui:include src="footer.xhtml"/>
</footer>
<script type="text/javascript" src="resources/js/bootstrap.js"></script>
</body>
</html>
```

Figure 4.4 : *An example of XHTML view page which includes other templates.*

4.3 Data import, parsing and storing to the internal data structure

Challenge task of the solution

The second task of implementation was to import data from the provided APIs of different learner models. It is done by reading JSON data from the provided restful web services of learner models. Based on the selected learner models, the VisuLeMo is fetching interests of a user and parsing them to Java objects using Google Gson. The challenge task of this thesis was to create an internal data structure, which can be used for mapping among the APIs of learner models. So, based on the requirements of learner models and after analyzing data from different learner models, I have created an internal data structure as shown in Figure 4.5. This data structure has been realized in the VisuLeMo by implementing the database tables as shown in Figure 4.6.

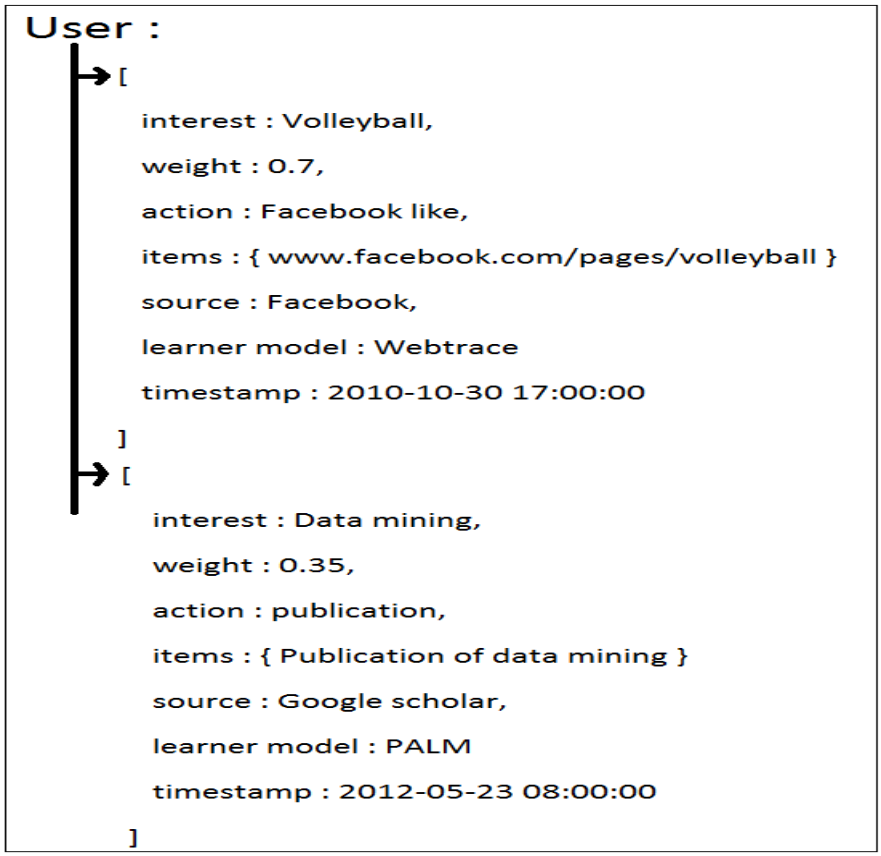


Figure 4.5 : *An internal data structure for storing interests from learner models with 2 examples.*

The terms used in the internal data structure are explained below:

Terms of the internal data structure

- **interest**: It refers to the actual name of an interest, in which a particular user is interested. For an example "Volleyball", if a user is interested in Volleyball.
- **weight**: It is also referred as 'Importance Value' or 'Score'. It can have any floating point value including or between 0 and 1. A value '1' represents that a user is highly interested and a value '0' represents that a user is very less interested.
- **action**: It represents a user action based on which a learner model comes to know about her interest. For an example "Facebook like". If a user has liked a Facebook page for Volleyball.
- **items**: One interest can have multiple items. An item represents the reference from where an interest comes from. For an example, a link to the Facebook page of Volleyball.
- **source**: It refers to the source name from where an interest comes from.

- **learner model**: It represents the learner model from which an interest has been imported.
- **timestamp**: It represents the modification time of an interest.

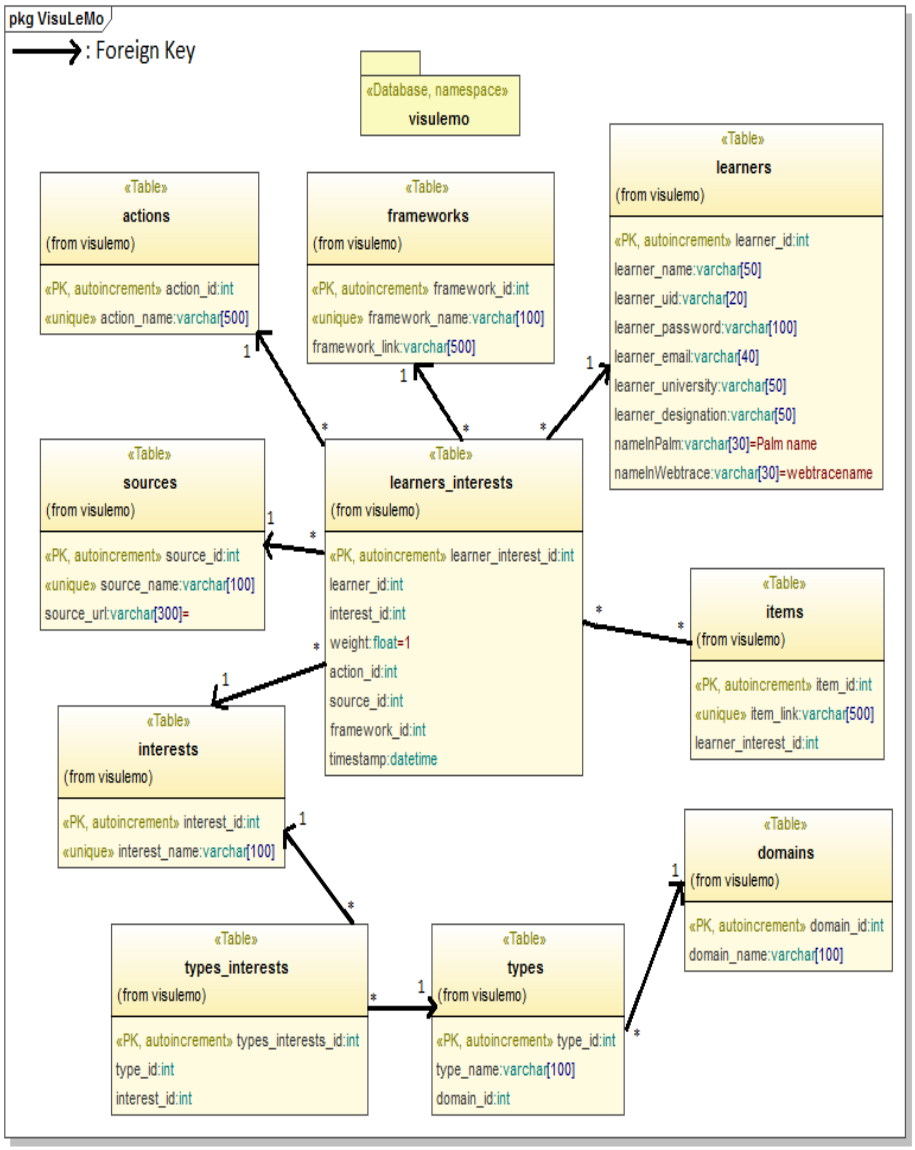


Figure 4.6 : *Database class diagram of the VisuLeMo with cardinality.*

Internally, mapping of the fetched data to the internal data structure has been done and stored in database. In the current version of VisuLeMo, it imports data from the PALM and the Webtrace. In section 4.3.1 and section 4.3.2, mapping of the PALM and the Webtrace data to the internal data structure have been explained. Adding more learner models for importing data is also possible and has been described in section 4.3.3.

4.3.1 Mapping of the internal data structure to the PALM API

Figure 4.7 shows an example JSON response string for the interests from PALM API. This string is parsed to a Java object using Google Gson and fields of it are mapped to the respective fields of the internal data structure as shown in Figure 4.8. For the field "action" and the field "source", there are no corresponding fields from the PALM API. The interests in PALM are extracted from academic papers of the corresponding user [1], so I kept the default value for action field as "publication". However, a user can add values for the fields "action" and "source" using VisuLeMo interface.

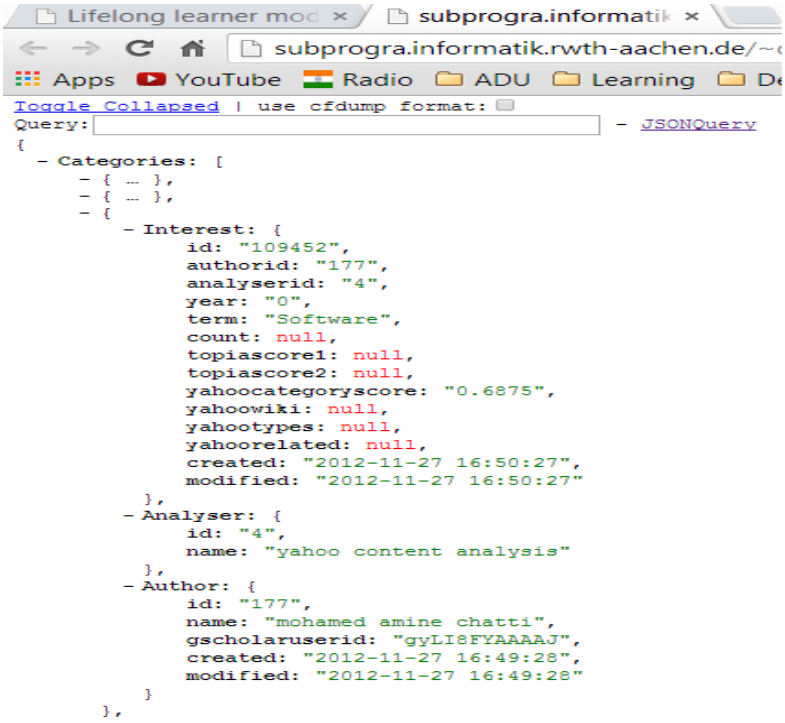


Figure 4.7 : *JSON response string from the PALM API for the interests.*

Internal data structure field	PALM API response field
interest	Categories.Interest.term
weight	Categories.Interest.yahoocategoryscore
action	No action, Default "publication"
items	Categories.Interest.yahoowiki
source	No source
learner model	Default "PALM"
timestamp	Categories.Interest.modified

Figure 4.8 : *Mapping of the internal data structure fields to the PALM API response fields.*

4.3.2 Mapping of the internal data structure to the Webtrace API

During the work of this thesis, Webtrace does not have any API which can provide access to its data from outside. However, the thesis work of Webtrace was done at Learning Technologies Research Group, RWTH Aachen University, where the work of VisuLeMo was running. So, I get access to the database of Webtrace to use it in VisuLeMo to fetch social interests of a learner gathered using Webtrace interface. Figure 4.9 shows mapping of the internal data structure fields with the Webtrace database fields. In Figure 4.9, the field value "cake_facebook_likes.name" represents "name" column of "cake_facebook_likes" table from Webtrace database and the same way other values can be interpreted. For the field "weight", there is no corresponding field from Webtrace database. The default value of "1.0" has been kept and it can be edited by a user using VisuLeMo interface.

Internal data structure field	Webtrace database field
interest	cake_facebook_likes.name or cake_lastfm_artists. name
weight	None, Default "1.0"
action	"Facebook like" or "Lastfm artist"
items	"http://www.facebook.com/"+ cake_facebook_likes.id or cake_lastfm_artists.url
source	"Facebook" or "Lastfm"
learner model	Default "Webtrace"
timestamp	cake_facebook_likes.modified or cake_lastfm_artists. modified

Figure 4.9 : *Mapping of the internal data structure fields to the Webtrace database fields.*

4.3.3 Adding new learner model

Extending the VisuLeMo by adding further learner models is possible. This can be done by following below steps:

Steps to add new learner model

Step 1: Decide mapping of data from new learner model to the internal data structure as it was shown for the PALM and the Webtrace in two previous sections.

Step 2: Fetch data from new learner, parse it to Java object and write code for the mapping that was decided in Step 1.

Here, I am explaining step 2 in more details with an example of PALM. In the APIJsonBean.java class of VisuLeMo, there is one method called frameworkSelection(). Inside this method, new condition for new learner model, the code for fetching data from new learner model and the code for parsing this data to Java object have to be added. Figure 4.10 shows frameworkSelection() method of APIJsonBean.java class with the condition for PALM. After parsing the fetched data to Java object, use this Java object to map with the values for internal data structure in editLearnerInterests.xhtml. Figure 4.11 shows this mapping for PALM.

```
107  public String frameworkSelection(String nameInPalm, String nameInWebtrace){
108
109      if((selectedFrameworks.size()==1) && selectedFrameworks.get(0).equals("PALM"))
110      {
111          //System.out.println("Selected learner model is PALM");
112          wbIs.clear();
113          try{
114          String palm_interests_URL="http://subprogra.informatik.rwth-aachen.de/~ddugosija/llmian/Api/interests/";
115          nameInPalm = nameInPalm.replaceAll(" ", "%20");
116          json = readUrl(palm_interests_URL+nameInPalm);
117
118          JsonObject jobj = parser.parse(json).getAsJsonObject();
119          ipage=gson.fromJson( jobj , InterestPage.class);
120
121          }catch(Exception e)
122          {
124              e.printStackTrace();
125          }
126          return "editLearnerInterests";
127
128      }else if((selectedFrameworks.size()==1) && selectedFrameworks.get(0).equals("webtrace")){
          //System.out.println("Selected learner model is Webtrace");
```

Fetching interests from PALM

Parsing it to Java object

Figure 4.10 : *frameworkSelection() method of APIJsonBean.java class.*


```
<c:forEach var="category" items="${apijsonbean.ipage.categories}">
<tr>
 Interest:
 <td><p:inplace >
      <p:inputText required="true" id="interest_${category.interest.id}" value="${category.interest.term}"/>
    </p:inplace>
 </td>
    <h:inputHidden id="learner_${category.interest.id}" value="${category.author.name}" />

 Learner Model:
 <td><h:outputText value="PALM"/></td>

 Weight:
 <td><p:inplace >
      <p:inputText required="true" id="weight_${category.interest.id}" value="${category.interest.yahoocategoryscore}"/>
    </p:inplace>
 </td>

 Action:
 <td><p:inplace >
      <p:inputText required="true" id="action_${category.interest.id}" value="publication"/>
    </p:inplace>
 </td>

 Items:
 <td><p:inplace >
      <p:inputText required="true" id="items_${category.interest.id}" value="${category.interest.yahoowiki}"/>
    </p:inplace>
 </td>

 Source:
 <td><p:inplace >
       <p:inputText required="true" id="source_${category.interest.id}" value="No source"/>
     </p:inplace>
 </td>

 Timestamp:
 <td><p:inplace >
      <p:inputText required="true" id="timestamp_${category.interest.id}" value="${category.interest.modified}"/>
    </p:inplace>
 </td>

</tr>
</c:forEach>
```

Figure 4.11 : *Mapping the fetched data values to internal data structure values.*

Step 3: Save the newly mapped values for internal data structure into database. In APIJsonBean.java class, there is one method called saveInterestsToDatabase(). Inside this method, there is already code for inserting PALM and Webtrace data into database. The code for inserting newly mapped values for internal data structure into database has to be added in this method.

Step 4: Add one more checkbox for new learner model in the learner model selection page (chooseFramework.xhtml) as shown in Figure 4.12. Based on the selection of checkbox here, the condition of frameworkSelection() method of the Step 2 has to be given.

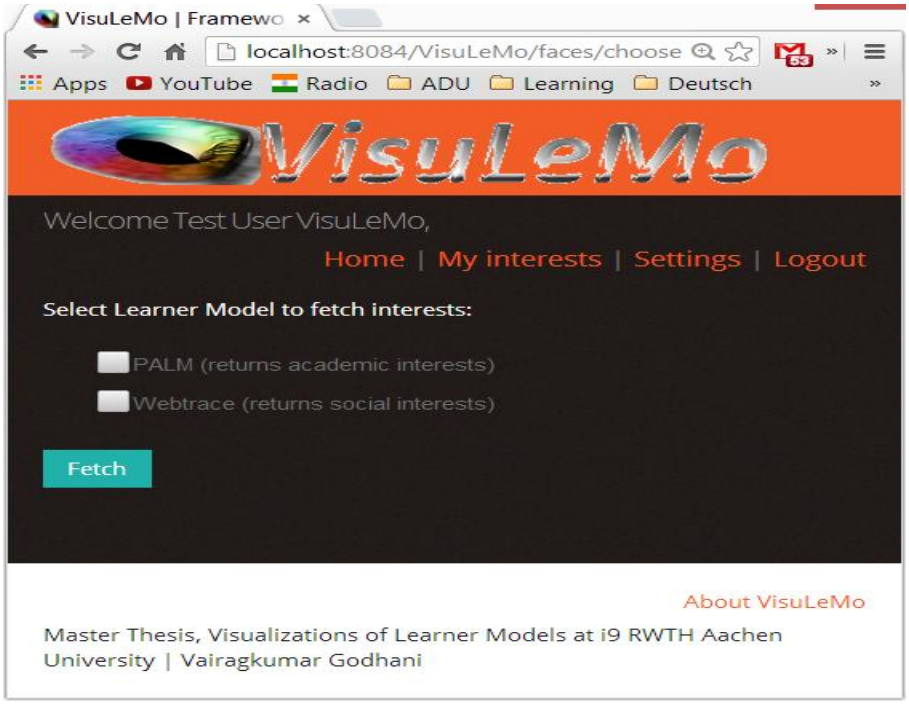


Figure 4.12 : *Learner model selection view page (chooseFramework.xhtml).*

After successfully following above 4 steps, new learner model should be added to the VisuLeMo.

4.4 Clustering based on Freebase

In section 3.3, it has been already explained about the Freebase API. In this section, how it has been implemented for clustering the fetched interests will be explained. In Freebase, every term is referred as a 'Topic'. A topic can come under several 'Types' and each 'Type' can come under a 'Domain'. For more understanding of "Topics", "Types" and "Domains" with an example, see section 3.3. For each interest, the VisuLeMo is fetching its "Types" and "Domains" from Freebase. In this thesis, at the time of fetching information from Freebase, I am referring an interest as a topic of the Freebase, "Types" of the Freebase as categories of an interest and "Domains" of the Freebase as parent categories of an interest. Figure 4.13 explains this terminology with an example.

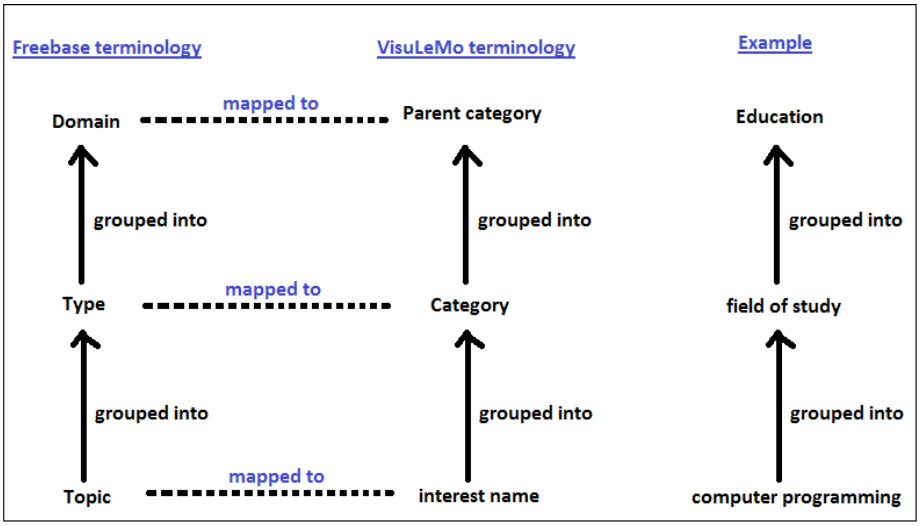


Figure 4.13 : *Mapping of Freebase to VisuLeMo terminology with an example.*

The URL used for the Freebase API query to fetch categories look like

```
https://www.googleapis.com/freebase/v1/search?query=volleyball&output=(type)&limit=1
```

where, volleyball is the queried interest for which query is requested to get categories from the Freebase. To know, what parameters like 'output=(type)' and 'limit=1' will do, see section 3.3. Figure 4.14 shows response from the Freebase for categories of 'Volleyball'. The response string for each interest is parsed to Java object using Google Gson. And then it will be stored into database, which is realized by database tables called 'types_interests', 'types', 'interests' and 'domains' (see Figure 4.6). Based on this information, groups of interests are made and used in visualizations.

```
status: "200 OK",
result: [
  - {
      mid: "/m/07_53",
      id: "/en/volleyball",
      name: "Volleyball",
    - notable: {
        name: "Sport",
        id: "/sports/sport"
      },
      lang: "en",
      score: 237.819763,
    - output: {
      - type: {
        - /type/object/type: [
            - { … },
            - { … },
            - {
                id: "/base/skosbase/topic",
                mid: "/m/0bmsftm",
                name: "Topic"
              },
            - {
                id: "/base/sportbase/sport",
                mid: "/m/04_k2j8",
                name: "Sport"
              },
            - { … },
            - {
                id: "/cvg/computer_game_subject",
                mid: "/m/05pvlrq",
                name: "Video Game Subject"
              },
            - { … },
            - {
                id: "/olympics/olympic_sport",
                mid: "/m/048n9_d",
                name: "Olympic discipline"
              },
            - {
                id: "/base/skosbase/vocabulary_equivalent_topic",
                mid: "/m/0h817y9",
                name: "Vocabulary Equivalent Topic"
              },
```


Figure 4.14 : *Response from the Freebase for categories of 'Volleyball'.*

4.5 Visualization

Visualization of interests is the core part of this thesis. Once interests have been imported from learner models and their categories have been fetched from the Freebase, they can be represented differently based to get important information out of it. This can help a user to understand her interests in a better way to promote self reflection and awareness. The VisuLeMo uses six chart types which are SpaceTree, HyperTree and Icicle from InfoVis Toolkit and Column chart, Pie chart and Bubble chart from Highcharts JS. Demo for these chart types for InfoVis Toolkit can be seen at [24] and for Highcharts JS at [25]. The chart types SpaceTree, HyperTree and Icicle are well suited for representing hierarchical information and in this case they best fit for "Cluster of interest categories" and "Weighted hierarchy of interests" visualizations. The visualizations that were created

to help a user to get important information out of her interests are explained in the following sections.

4.5.1 "Importance of interests" visualization

This visualization represents interest names versus their importance values (weights) as a Column chart. As shown in Figure 4.15, on the x-axis, there are interest names and on the y-axis, there are corresponding importance values. From this, a user can see which interest has how much importance value and can easily compare the importance of different interests based on the bars in chart. If a user has several interests and she wants to see only 'TOP 5' interests with the highest importance value then it is possible with the provided filter. On mouse over of each bar will show tooltip explaining the selected bar. Exporting and downloading the chart as PNG image, JPEG image, PDF document or SVG vector image is possible. The legends of chart represent learner models from where the interests come from. The bars for interests from the same learner model are represented in same color.

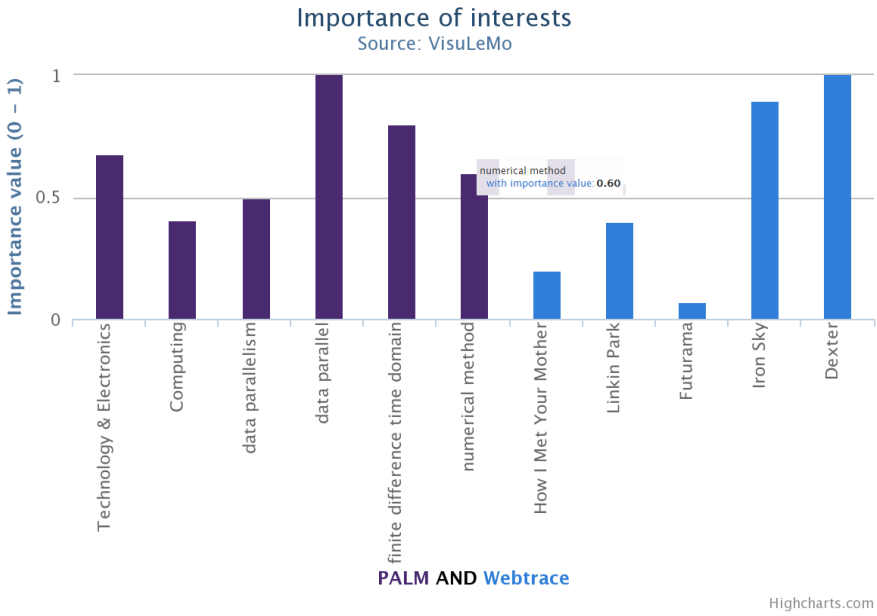


Figure 4.15 : *"Importance of interests" visualization.*

4.5.2 "Interest share" visualization

This visualization represents % share of each interest compare to all interests as a Pie chart. The information which a user will get from this visualization is what % of all interests is covered by each interest. On mouse over to the portion of interest will show tooltip telling the % of that interest. The same visualization is also available for top 5 interests with the highest importance value. In legends at the bottom of chart, there is filter

based on each interest. If a legend has been clicked, it will remove corresponding interest from the chart and clicking again will add it back. This filter can be useful, if a user wants to see the share of selected interests. Exporting and downloading the chart as PNG image, JPEG image, PDF document or SVG vector image is possible.

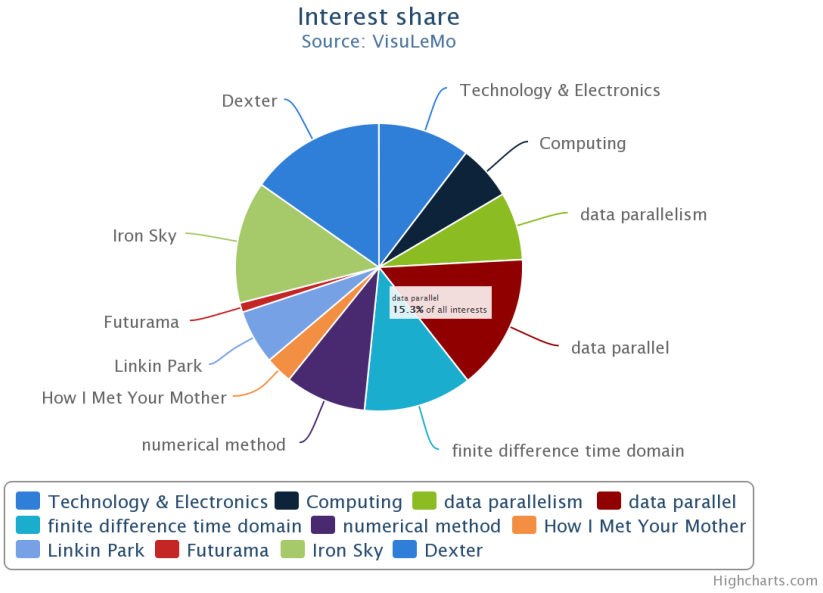


Figure 4.16 : *"Interest share" visualization.*

4.5.3 "Evaluation of interests over time" visualization

This visualization is created using 3-dimensional bubble chart with interest names on the x-axis, modification times on the y-axis, and importance values of interests as radius of bubble (See Figure 4.17). The more importance value of interest, the bigger bubble it will have. This visualization shows evaluation of interests over time. For an example, if an importance value of one interest was 0.3 before six months and now it has been modified to 0.75. In visualization, it will be shown that the bubble for that interest was smaller before six months and now it has become bigger because of the modification. In short, it tells that a user is more interested in that interest compare to six months ago. To filter visualization for top 5 interests' evaluation is possible. Exporting and downloading the chart as PNG image, JPEG image, PDF document or SVG vector image is also possible.

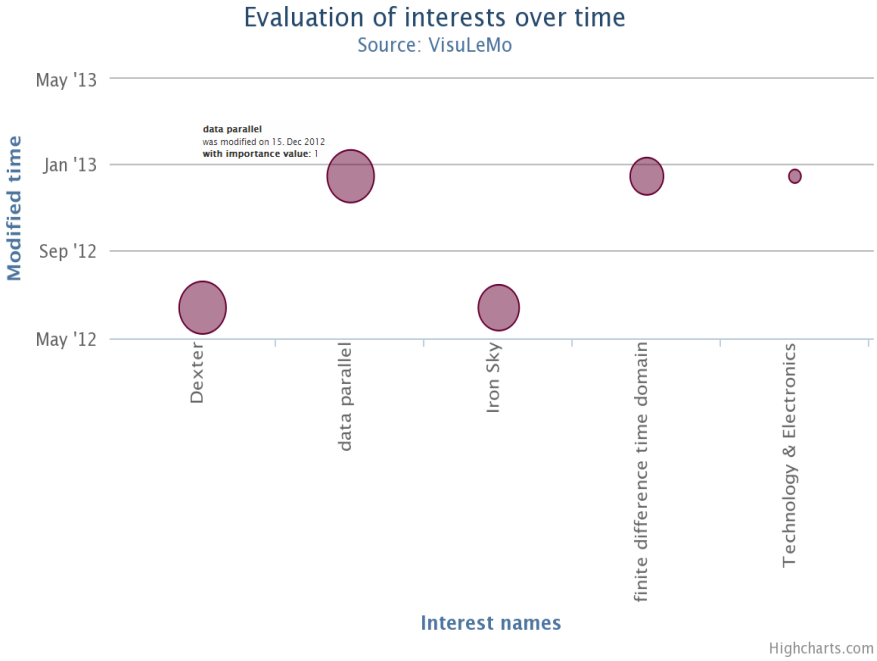


Figure 4.17 : *"Evaluation of interests over time" visualization.*

4.5.4 "Cluster of interest categories" visualization

This visualization is specially created to represent categories of interests that are fetched from Freebase. The "Cluster of interest categories" visualization is using HyperTree chart type. As shown in Figure 4.18, in the center of visualization it displays a user name, as a parent branch node it shows parent category of interest , afterwards as the sub-branch node it display category of interest and as the leaf of branch it shows interest name. With this visualization, a user can easily see which interests come under the same category or parent category. A user can also see from which different areas her interests are coming. A left click on node will center the clicked node and adjust other parts of visualization. It also supports advanced options such a panning the canvas by dragging mouse and zooming graph with the help of mouse wheel. Filtering this visualization for top 5 interests with the highest importance is also provided.

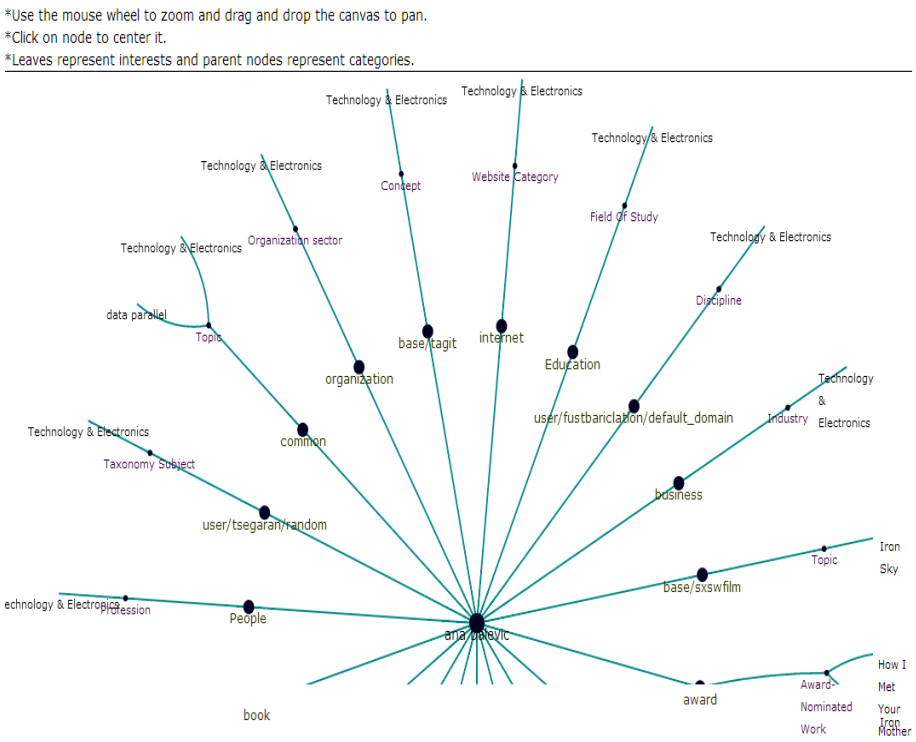


Figure 4.18 : *"Cluster of interest categories" visualization.*

4.5.5 "Weighted hierarchy of interests" visualization

The "Weighted hierarchy of interests" visualization is using SpaceTree chart type. As shown in Figure 4.19, interests and their categories are represented as hierarchy of nodes. An important thing in this visualization is calculation of background color for interest node based on the number of categories under which an interest comes. If an interest comes under more number of categories, background color of the node representing this interest will be darker and if an interest comes under less number of categories, background color of the node representing interest will be lighter. This is why the name is given as weighted hierarchy of interests. Clicking on interest node will display categories under which the clicked interest comes. In this visualization, a user can easily compare which interest has more or less categories based on lighter or darker background color of interest node. It also supports advance options such as panning the canvas by dragging mouse and zooming graph with the help of mouse wheel.

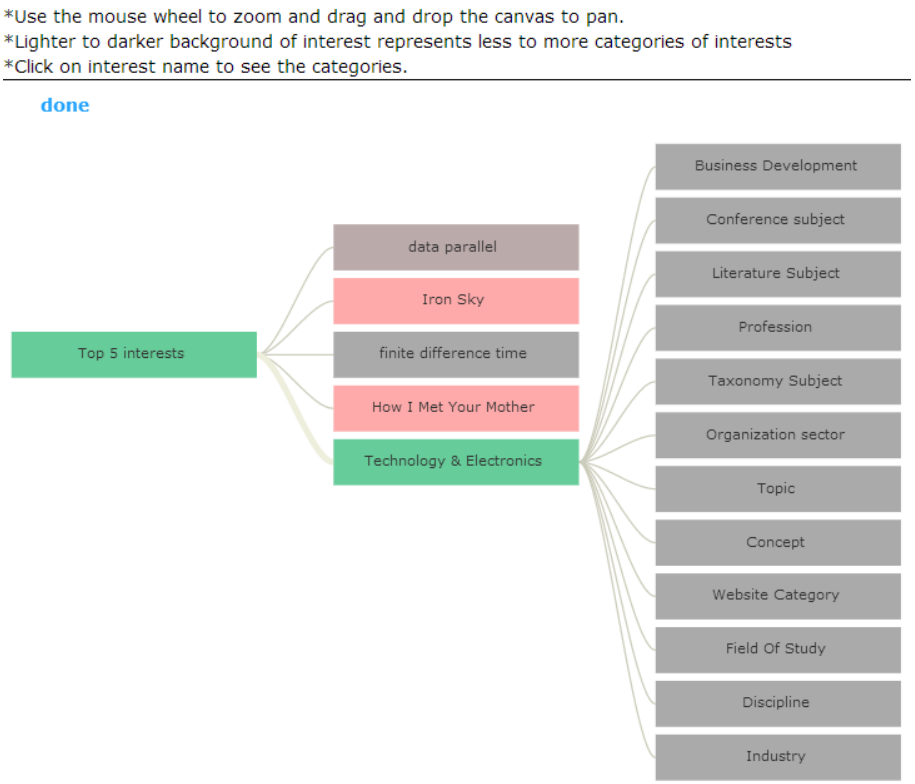


Figure 4.19 : *"Weighted hierarchy of interests" visualization.*

In sum, with the above specified implementation for the solution of this thesis, the VisuLeMo had enough results as it was planned. So, I have decided to do user evaluation to improve the current results and make improvements as per user feedback which is explained in the next chapter.

Chapter 5 Evaluation

This chapter shows the evaluation of the work based on user interviews and later illustrates set of improvements which are made based on user suggestions.

5.1 DIA cycle and addressed evaluation questions

DIA cycle

The development of VisuLeMo was done by using the Design-Implement-Analyze (DIA)[60] cycle of software development. In the DIA cycle, initially a design of framework is made, then it will be implemented and afterwards it will be evaluated. Based on evaluations of previous prototype, new design will be made and again it will be passed through the phases of Implementation and Analysis. The cycle of these 3 phases will run until the required software is developed.

Evolution of VisuLeMo

In the VisuLeMo, I have used the DIA cycle with 3 repetitions. The first version was for creating the mockup of VisuLeMo using Basalmiq[61]. Appendix B shows screenshots of this mockup. It includes the basic GUI[62] of VisuLeMo, an idea of importing data from learner models and visualization techniques. I have shown it and have taken suggestions from 4 students doing master degree at RWTH Aachen University and also from the supervisors of this thesis. Keeping those suggestions in mind, I build an actual running JSF application with basic functionalities. This version of VisuLeMo contained functionalities like user registration, authentication, fetching learner data from the PALM and modifying them. To make the final version of VisuLeMo, I made improvements such as adding second learner model called Webtrace, adding CSS for good GUI by creating templates in view pages, supporting of Bootstrap for responsiveness, grouping of interests based on Freebase and different visualizations. Appendix C shows screenshots of the final version of VisuLeMo.

Participants of user interviews

An evaluation of the final version of VisuLeMo has been conducted in two sets of user interviews by giving a chance to a user to use the VisuLeMo and taking user's feedback. Each set was carried out with 3 test users. From these total 6 test users, 4 users were studying masters in Computer science

[60] http://www.yovisto.com/video/19448
[61] http://balsamiq.com/
[62] http://en.wikipedia.org/wiki/Graphical_user_interface

at RWTH Aachen University[63], 1 user was studying Mechanical engineering at RWTH Aachen University and 1 user was studying Medial science at Uniklinik RWTH Aachen[64]. Out of 6 participants, 1 was female participant and 5 were male participants. During user interviews, users had to perform the tasks using VisuLeMo. Section 5.2 describes the tasks those were given to users during evaluations. Section 5.3 lists out improvements which were made to the VisuLeMo based on the user feedbacks. In evaluation of the VisuLeMo, the following questions were addressed:

Addressed evaluation questions

- How is usability of the VisuLeMo?
- Does it have good interface to import data from different learner models?
- From the provided visualizations: Which are the good visualizations?
- How is support for awareness and self-reflection?

5.2 Tasks given to a user during evaluation

Introduction given to participants

In the beginning of a user interview, a brief introduction about VisuLeMo was given to a user, which was as follows: "A person can have interests from various fields such as social, education, entertainment, etc. From such details of a person, we can create a person (user)'s model called learner model. There are already developed frameworks (called learner models) such as

- PALM: This gives academic interests of a user.
- Webtrace: This gives social interests of a user.
- And other frameworks like these can be developed and added.

So, the VisuLeMo will fetch user's interests from different frameworks as mentioned above and visualize them. And the aim is to see whether these visualizations help user to promote awareness and self-reflection".

After introduction, the VisuLeMo was shown to the users and were requested to perform below tasks. During these tasks, the users were asked simple questions such as to tell what they think clicking on particular link or button will do and notes to improve the VisuLeMo were taken. After completion of the tasks, the users were asked if they can suggest any improvements in the VisuLeMo.

[63] http://www.rwth-aachen.de/
[64] http://www.ukaachen.de/

Tasks for the user evaluation

Task 1: Create new user, login to the VisuLeMo and update your ID for the learner model PALM from the settings.

Task 2: Fetch your interests from the Webtrace and see importance value of your interests in the visualization.

Task 3: Update the importance value/weight of 2 interests from already fetched interests and see interest share of these 2 interests in the visualization.

Task 4: View categories and parent categories of 1 already fetched interest after that check in the visualization that which other interests come under the same category as this interest.

Results of the user interviews

Figure 5.1 shows the overview of user interviews for the VisuLeMo. The duration mentioned is only the duration of performing the core task. It does not include the discussion with the user. Duration of each task was measured separately and finally added to get total duration. Actual duration of each full interview was on an average of 30-40 minutes.

	Gender	Duration	Status
Test Set 1:			
Person 1	Male	7:40 min	Computer science student
Person 2	Male	12:10 min	Medical student
Person 3	Male	8:30 min	Computer science student
Test Set 2:			
Person 4	Male	8:40 min	Mechanical engineering student
Person 5	Female	7: 50 min	Computer science student
Person 6	Male	8:10 min	Computer science student

Figure 5.1 : *Overview of the user interviews.*

5.3 Improvements based on evaluation

User interviews with the users were very beneficial. During the user tests, I got various suggestions to improve the VisuLeMo. Below are the listed improvements that were made based on user feedback.

5.3.1 Improvements made after first set of user tests

Improvements related to usability:

- **Changed links and button names** for clear understanding. Example 1 - "Go to steps" link name to "Fetch interests" link name with title tool-tip explaining what clicking on link will do. Example 2 - "Different Visualizations" button name to "Visualize" button name.
- **Added validations** in form for the required fields and used PrimeFaces messages to give user feedback.
- **Combined basic visualization and category visualization page** to one visualization page. Previously, there were 2 pages for visualizations. One was for basic visualization, where all the charts were generated using HighCharts JS. Another one was for category visualizations, where the charts were generated using InfoVis Toolkit. The reason for keeping them separate was displaying issue of two different chart library graphs in a single page. InfoVis Toolkit and HighCharts have their base CSS and JS files, where some styling codes were overridden by each other resulting in not displaying graphs from these 2 JS libraries in a single page. Now these files are adjusted, so that it is working correctly (See Figure 5.3).
- **Added "i (info)" icon to the terms, which need more explanation.** For an example, the term "Importance value" was not understandable to the users during interviews. So to add its definition "i (info)" icon is added. Now on hovering over the "i (info)" icon, it provides explanation of the term (See Figure 5.2).
- **Changed the label names.** Previously, everywhere a name "learner" was used, keeping in mind the context of learner model. For an example, in the registration form, the labels used were like Learner Name, Learner Id, Learner Password, etc. It was difficult for normal users to understand the context. So, these labels have been changed to "User" such as User Name, User Id, and User Password etc.
- **Changed Font colors and background colors of the tables.** Previously, contents inside tables were less readable, so to increase readability this improvement has been made (See Figure 5.2).

Improvements related to support of importing data from learner models:

- **Editing interests directly after fetching from learner models is removed.** This improvement is related to an evaluation question, how is support for importing data from learner models. According

to test users, they did not prefer to edit the fetched interests directly after fetching from learner models. They preferred to edit interests later in "My Interests" tab.

- **Improved the old time consuming flow of importing interests and collecting their Freebase categories.** Previously, first interests from learner models were imported, mapped to the internal data structure and then stored into database. Afterwards on the next view page, all interests were queried from database and their categories were requested from the Freebase and stored in the database. This whole flow was time consuming because of more data operations. The new flow is created such that each interest's categories are requested before storing into database. Once the categories are ready then all details of interests are inserted together into database.
- **Removed category view page of an interest after fetching the categories from Freebase.** Previously, categories of an interest were shown to user after fetching them from Freebase. User suggestion was to remove category view page and go directly to visualization page.

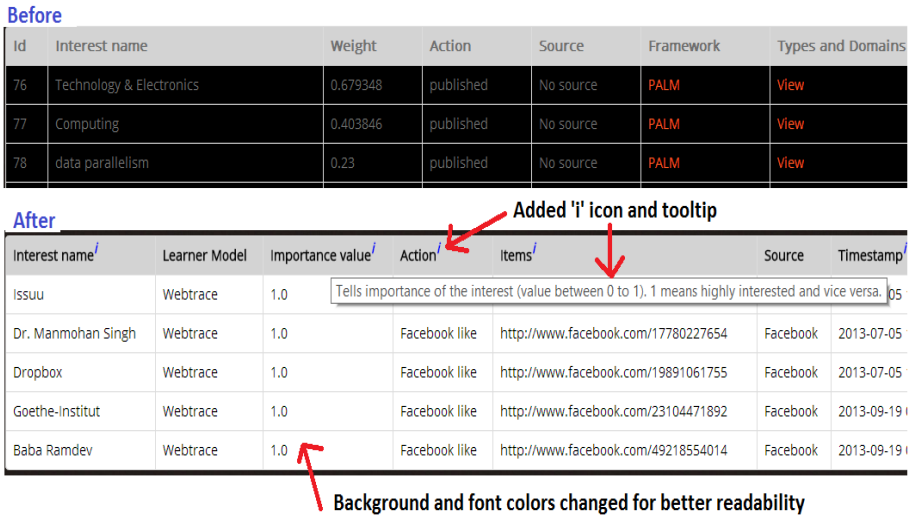


Figure 5.2 : *Sample improvement 1 based on user feedbacks.*

Improvements related to support of visualization, awareness and self-reflection:

- **In Pie chart, removed clicking on the portion.** Previously, clicking on the portion of a Pie chart was just expanding that portion. Information about the portion was already shown to user when

hovering over portion. Clicking on the portion was misleading, so it has been removed.

- **Category based visualizations were time consuming.** It has been improved by improving SQL queries and removing all system echo messages at the time of representing visualization.
- **Changed names of visualizations** from names of chart types to the names which describe which information a user will get from the visualization. For an example, from the names "Column Chart" and "Pie Chart" to the new names "Importance of the interests" and "Interest share" (See Figure 5.3).
- **Changed font colors and background inside charts** for making its content more readable.
- **Removed charts that were representing the same information but in different chart type**. For an example, "Importance of the interests" visualization with the Column chart and with the Dot chart will give the same information, but just in different format. In evaluation, user feedback was that different chart types do not make sense if they represent the same information.

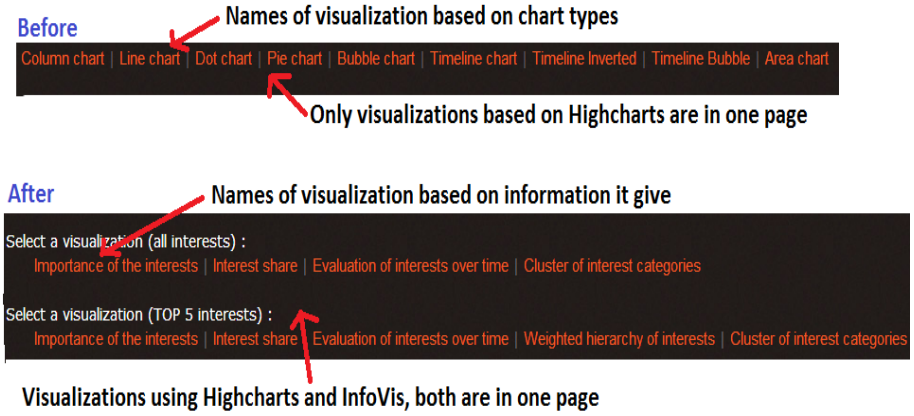


Figure 5.3 : *Sample improvement 2 based on user feedbacks.*

5.3.2 Improvements made after second set of user tests

Improvements related to usability:

- **Home link issue** on the logo of VisuLeMo. Previously, there was no active session condition check on the link of logo. If a user is logged in and clicks on Home logo, it should go to Home page, rather than Login page. This issue has been resolved.
- **Increased the font size** of navigation items and also added 'Home' item to the navigation items (See Figure 5.4).

- **In "My Interest" tab, added two links** for "Fetch Interests" and "Visualizations" (See Figure 5.4).
- **Removed social icons** from the footer and add "About VisuLeMo" page link in the footer (See Figure 5.4).

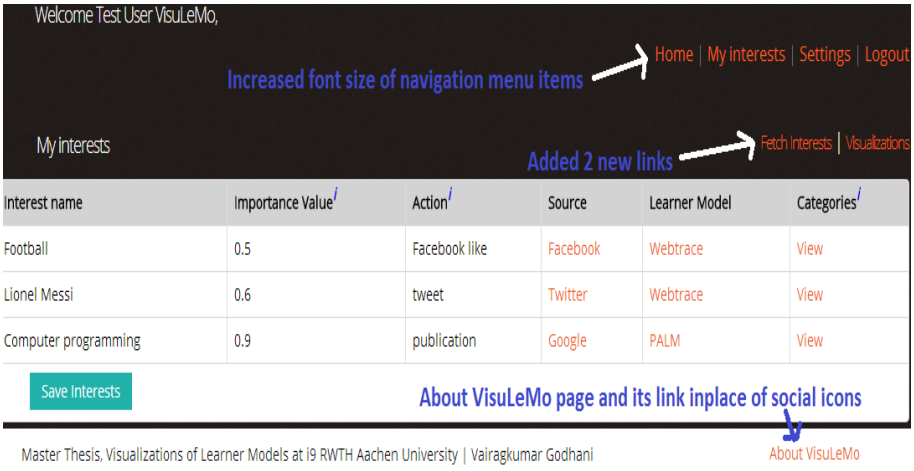


Figure 5.4 : *Sample improvement 3 based on user feedbacks.*

Improvements related to support of visualization, awareness and self-reflection:

- **Updated appropriate names and tool tips with in the charts**
- **Added zooming and panning functionalities for category charts.** Previous, it was hard to see visualization for large number of categories. With the help of zooming and panning functionalities, this has become easier.
- **Hints for visualizations** on the top of a chart, telling possible interaction within chart. For an example, what left or right click will do, how to do panning and zooming, etc. (See Figure 5.5).
- **Filter for viewing only top 5 interests with the highest importance value is added in each visualization.**

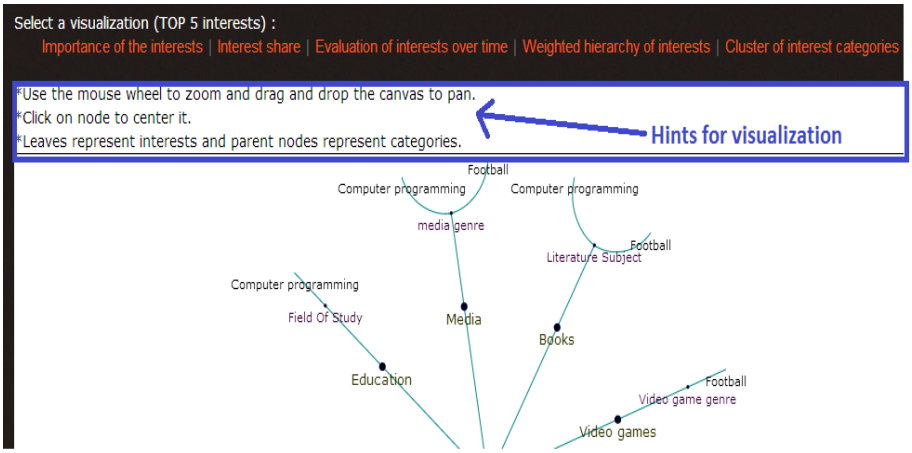


Figure 5.5 : *Sample improvement 4 based on user feedbacks.*

Summary of evaluation

In Sum, after evaluations and improvements, the user interface of VisuLeMo has become more self-describing. The support for importing data from learner models is easily usable and good. From evaluations, it has become clear that visualizations will represent interests in a better way, however whether they provide self-reflection and awareness mainly depend on, how relevant interests' data are. The VisuLeMo fetches interest data from the PALM and the Webtrace. If the fetched data is highly relevant to the user, then visualization of this highly relevant data helps to promote self-reflection and awareness.

Chapter 6 Conclusion and Future Work

This chapter summarizes the work done during this thesis by stating what results I achieved as a conclusion. At the end of this chapter, it tells possible ways to extend and improve functionalities of the VisuLeMo in future.

6.1 Conclusion

During this thesis, I developed a JSF web-application called the VisuLeMo. It allows to import user interests from two learner models named the PALM and the Webtrace. Adding more learner models is also possible. After fetching interests, it automatically collects these interests' categories/types and parent categories/domains from the Freebase API. It creates grouping of interests based on Freebase and visualizes them. Parent categories and categories fetched from the Freebase can also be viewed separately for each interest. The VisuLeMo also supports open learner modeling by allowing a user to update her interests if she doesn't aggree with some of her interest's data which were fetched from learner models. It also gives basic services such as user registration, user authentication, updating personal details and configuring user's name for the PALM and the Webtrace from the settings tab. Based on the interest data, the VisuLeMo also creates different visualizations such as "Importance of the interests", "Interest share", "Evaluation of interests over time", "Cluster of interests" and "Category hierarchy of TOP5 interests" to promote self reflection and awareness of a learner.

The evaluation showed that

- During evaluation phases, the user interface has been improved leading to a selfexplainatory work flow of the VisuLeMo.
- The user interface is reponsive, which enables the usase of VisuLeMo in various mobile devices and tablets.
- Importing user interests from two learner models is easy to use. The integration of further learner models is also possible.
- Data fetched from learner models are enriched with types and domains from the Freebase, which helps to create grouping of interests.
- It supports open learner modeling.
- It provides different visualizations of interests to promote self reflection and awareness of a learner.

As mentioned in the section 1.2, there are various challenges for the roles of lifelong learner modeling. The VisuLeMo is just one step forward in the direction to tackle these challenges. In sum, VisuLeMo should become aggregation point for the information of learner using data from different learner models to help realize learner's long term goals.

6.2 Future Work

In the following section, I am describing possible ways to future work that can be done to extend the functionalities of VisuLeMo.

6.2.1 Learner profile completion by adding more learner models

In the current version of VisuLeMo, it imports only learners' academic and social interests from the PALM and the Webtrace. However, a learner can have other interests, which are not covered by the PALM and the Webtrace. In section 3.2.3, it is explained how a new learner model can be added to the VisuLeMo. By adding more learner models, it will have more details about learner's interests, which makes the learner profile stronger. And it will help for better learner awareness and self-reflection.

6.2.2 Updating data from different learner models

In the current version of VisuLeMo, it fetches learner data from learner models such as the PALM and the Webtrace. The VisuLeMo also provides open learner modeling, so that a learner is able to modify her data inside VisuLeMo after it has been fetched. Now, consider a situation in which a learner has modified her data inside VisuLeMo and it has been saved. And on the other hand, there are new updates in the learner data in learner models like the PALM from where the VisuLeMo has previously fetched data and it has been updated by the learner. The VisuLeMo can be improved such that it provides the functionality which will update the learner's data of VisuLeMo keeping the changes made by a learner and also taking updates from learner models.

6.2.3 Integration of other semantic knowledge bases

The current version of VisuLeMo uses the Freebase API for grouping interests of a learner. Based on the clustering of interests according to the Freebase, visualizations are displayed. In the Freebase, grouping interests depends on types and domains under which a particular interest comes. In section 3.3.1 and 3.3.2, the Wikipedia and YAGO semantic knowledge bases are described. The functionality of VisuLeMo can be extended for clustering the interests based on other semantic knowledge bases such as Wikipedia

or YAGO. This will enable learner to see visualization of interests which will be based on clustering of other semantic databases.

Chapter 7 Bibliography

[1] Darko Dugošija, M. A. Chatti and Hendrik Thüs. Lifelong Learner Modeling in Academic Networks, Learning Technologies Research Group, RWTH Aachen University, Germany Oct. 2012.

[2] J. Kay and B. Kummerfeld. Lifelong Learner Modeling, Australian Research Council Projects DP0774532, DP0877665.

[3] Marius Klein, M. A. Chatti and Hendrik Thüs. Lifelong Learner Modeling in Social Networks, Learning Technologies Research Group, RWTH Aachen University, Germany.

[4] S. Bull and J. Kay. Student Models that Invite the Learner In: The SMILI():Open Learner Modeling Framework. International Journal of Artificial Intelligence in Education, 17(2), 89-120, 2007.

[5] DynMap+: Rueda, U., Larrañaga, M., Arruarte, A. & Elorriaga, J. A. (2006). A Concept Mapping Approach to Visualize Group Student Models, in W. Nejdl & K. Tochtermann (eds), Innovative Approaches for Learning and Knowledge Sharing, First European Conference on Technology Enhanced Learning, Springer-Verlag, Berlin Heidelberg, 383-397.

[6] QuizeGuide: Brusilovsky, P. & Sosnovsky, S. (2005). Engaging Students to Work with Self-Assessment Questions: A study of two approaches, Proceedings of 10th Annual Conference on Innovation and Technology in Computer Science Education, ACM Press, 251-255.

[7] Mr Collins: Bull, S. & Pain, H. (1995). 'Did I Say What I Think I Said, And Do You Agree With Me?': Inspecting and Questioning the Student Model, in J. Greer (ed), Proceedings of World Conference on Artificial Intelligence and Education, AACE, Charlottesville VA, 501-508.

[8] C-POLMILE: Bull, S. & McEvoy, A.T. (2003). An Intelligent Learning Environment with an Open Learner Model for the Desktop PC and Pocket PC, in U. Hoppe, F. Verdejo & J. kay (eds), Artificial Intelligence in Education, IOS Press, Amsterdam, 389-391.

[9] WILLOW: Perez-Marin, D.; Pascual-Nieto, I. & Rodriguez, P. (2009). Adaptive Computer Assisted Assessment, in Fu Lee Wang, Joseph Fong & Reggie Kwan (eds), Handbook of Research on Hybrid Learning Models: Advanced Tools, Technologies and Applications, Information Science Reference (antes Idea Group Reference), Portugal.

[10] ViSMod: Zapata-Rivera, J.D. & Greer, J. (2004). Interacting with Inspectable Bayesian Student Models, International Journal of Artificial Intelligence in Education 14(2), 127-163.

[11] PALM API, 2012 (visited on 07.01.2014), URL: *http://subprogra.informatik.rwth-aachen.de/~ddugosija/llmian/*.

[12] PALM API response for the interests of an example author, (visited on 07.01.2014). URL: *http://subprogra.informatik.rwth-aachen.de/~ddugosija/llmian/Api/interests/mohamed%20amine%20chatti*.

[13] Bootstrap Documentations, 2013 (visited on 07.01.2014), URL: *http://getbootstrap.com/getting-started/*.

[14] Apache Tomcat, 2014 (visited on 07.01.2014), URL: *http://tomcat.apache.org/*.

[15] Google Gson, a Java library (visited on 07.01.2014), URL: *https://code.google.com/p/google-gson/*.

[16] JavaScript InfoVis Toolkit Documentations (visited 02.01.2014), URL: *http://philogb.github.io/jit/static/v20/Docs/files/Core/Core-js.html*.

[17] Highcharts JS Documentations (visited on 07.01.2014), URL: *http://www.highcharts.com/docs/*.

[18] A running Webtrace application (visited on 12.01.2014), URL: *http://subprogra.informatik.rwth-aachen.de/~mklein/cake/*.

[19] Freebase API (visited on 12.01.2014), URL: *https://developers.google.com/freebase/*.

[20] Perez, D., A. Gliozzo, E. Alfonseca, C. Strapparava, B. Magnini, and P. Rodríguez. About the effects of combining Latent Semantic Analysis with other Natural Language Processing techniques for free-text assessment. International Journal Signos, 38(59):325-343. (2005).

[21] Gerhard Fischer. User modeling in human–computer interaction. User modeling and user-adapted interaction, 2001.

[22] Examples of learner models (visited on 07.01.2014), URL: *http://www.eee.bham.ac.uk/bull/lemore/examples.html*.

[23] Codester HTML-CSS template (visited on 07.01.2014), URL: *http://templatecreme.com/list/codester.*

[24] InfoVis Toolkit Demo (visited on 07.01.2014), URL: *http://philogb.github.io/jit/demos.html.*

[25] Highcharts Demo (visited on 07.01.2014), URL: *http://www.highcharts.com/demo.*

[26] D. Rosenberg and M. Stephens. Use Case Driven Object Modeling With UML: Theory And Practice. Apresspod Series. Apress, 2007. ISBN: 9781590597743.

Appendix A: Compiling the VisuLeMo

Note: In the CD, you will find project zip file, all the prerequisites mentioned below, 2 SQL files and thesis paper's PDF file.

Prerequisites:

- Netbeans IDE (Framework was developed in Netbeans version 7.3.1)
- Java Development Kit (JDK)
- MySQL (Import 2 SQL files: visulemo.sql and webtrace.sql)
- Apache Tomcat server (Used version was 7.0.34.0)
- PrimeFaces library 3.5 (primefaces-3.5.jar)
- Google Gson library 2.2.4 (gson-2.2.4.jar)
- JSF (javax.faces-2.2.1.jar)
- JSTL (jstl-1.1.jar)
- mysql-connector-java-5.1.24-bin.jar
- commons-fileupload-1.3.jar

In order to use the framework, follow the steps mentioned below:

1) First of all, import 2 SQL files: visulemo.sql and webtrace.sql in your MySQL database.
    - If you don't have SQL database, you can install XAMMP from the CD and use PhpMyAdmin of XAMMP for SQL database to import these 2 files.

2) Import project zip file in Netbeans IDE:
    - File -> Import Project -> From zip

3) Inside the Project, check library dependency paths. Normally all paths should be fine. If any path is unlinked then provide correct path for the libraries listed in prerequisites.
    - In Project Explorer of Netbeans. Right click Libraries -> Properties. Here, you can add libraries if some libraries are missing.

4) Specify database username, password and database name in the DbConnection.java file.

- Default username is "root" and password is ""
- Default databases names are "visulemo" and "webtrace".

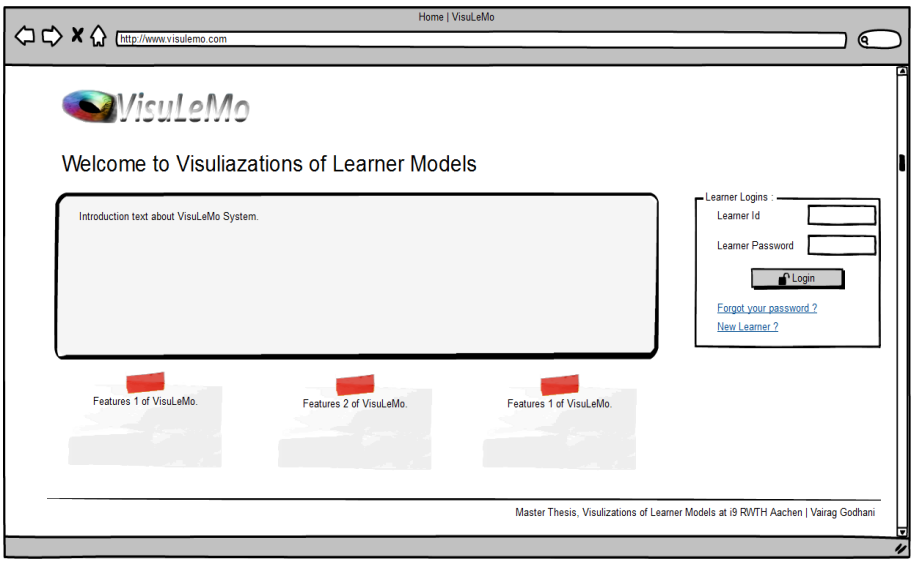
5) Build the project and run it on the Apache Tomcat server.
   - If you don't have your data in PALM and Webtrace, You can use the test account created for the "Test User VisuLeMo". Both Login Id and Password is test_visulemo.

Have a fun playing with VisuLeMo!

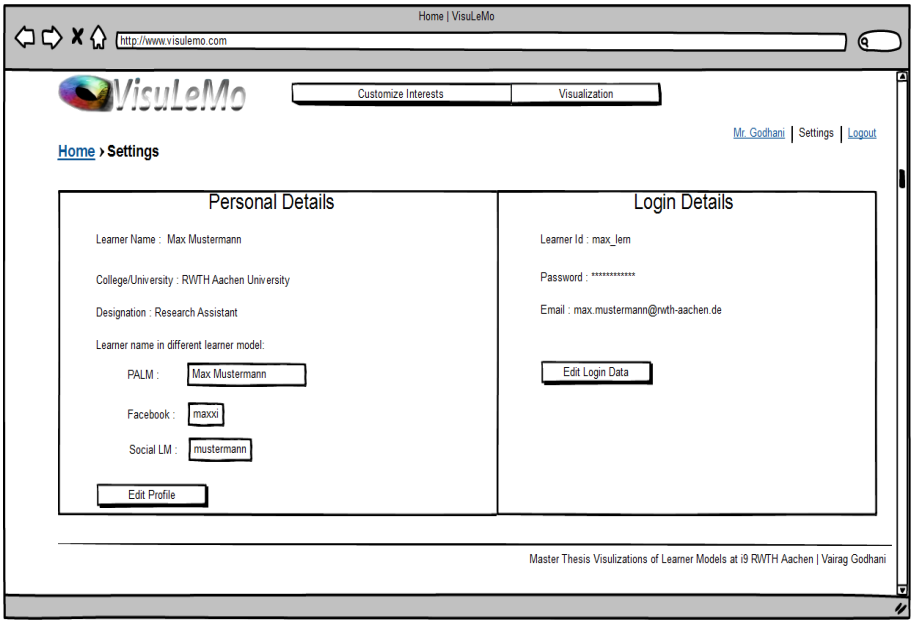
Suggestions, feedbacks, questions and error reports are always welcome and can be e-mailed to vairagkumar.godhani@rwth-aachen.de

Appendix B: Balsamiq mockup screenshots

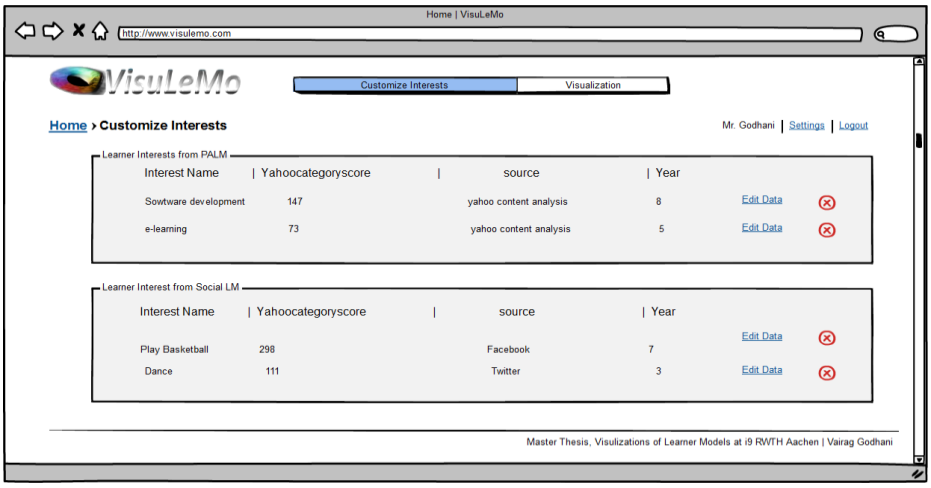
a) Login page of the VisuLeMo.



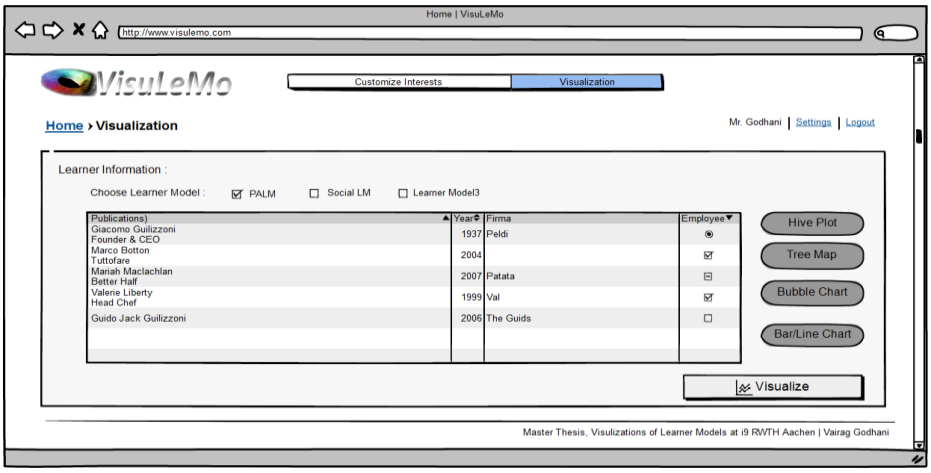
b) User settings page of the VisuLeMo.



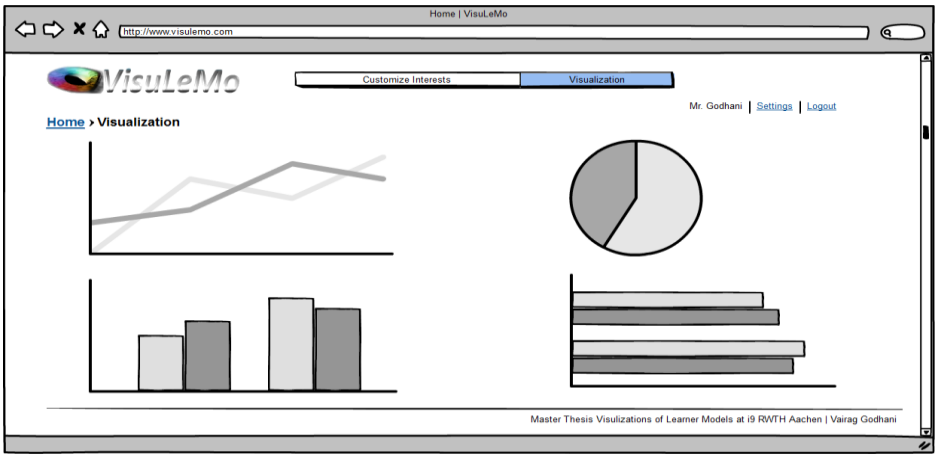
C) Customize interest page (which is later referred as 'My Interests' page).



d) Choosing different visualizations page.

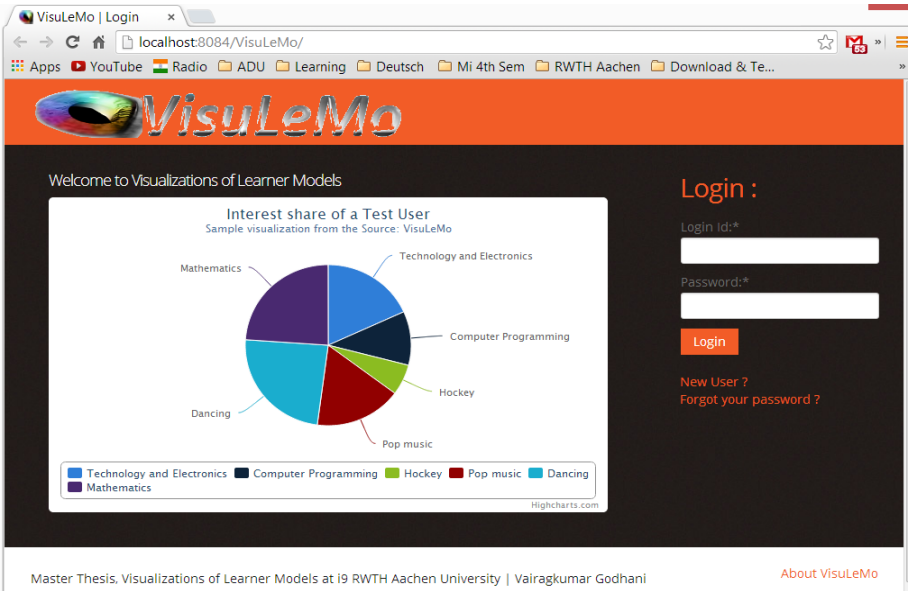


e) Sample visualization page where different visualizations will be shown.

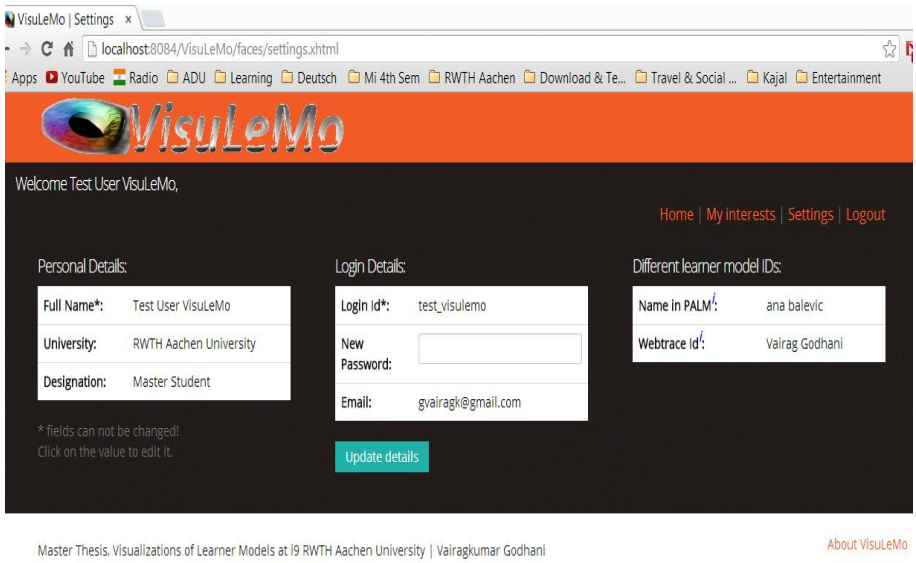


Appendix C: Final VisuLeMo JSF application screenshots

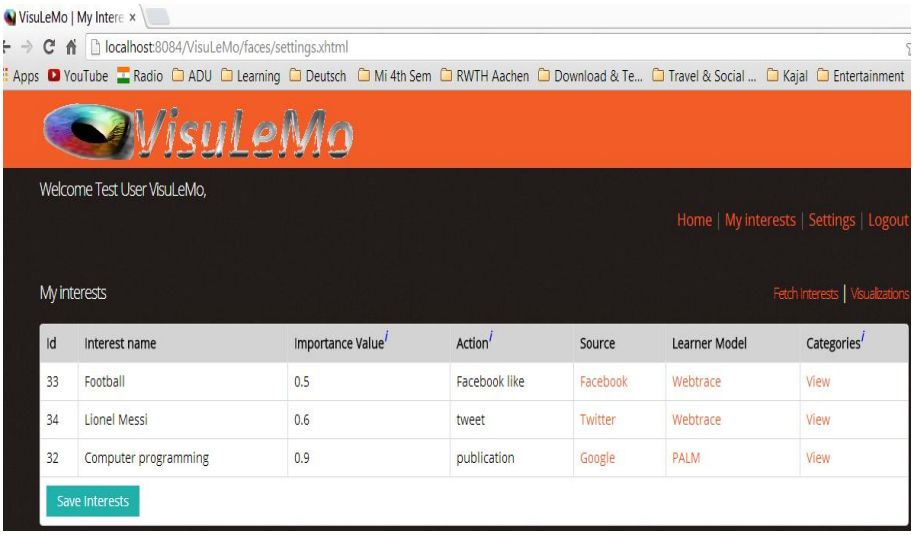
a) Login page of the VisuLeMo.



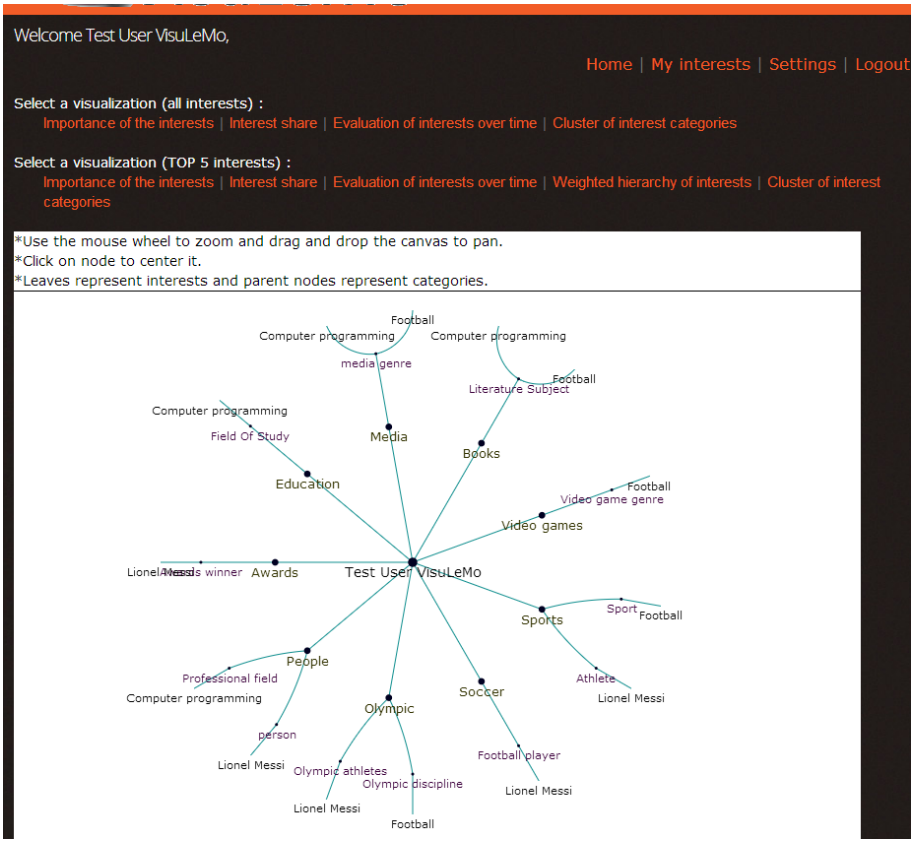
b) User settings page of the VisuLeMo.



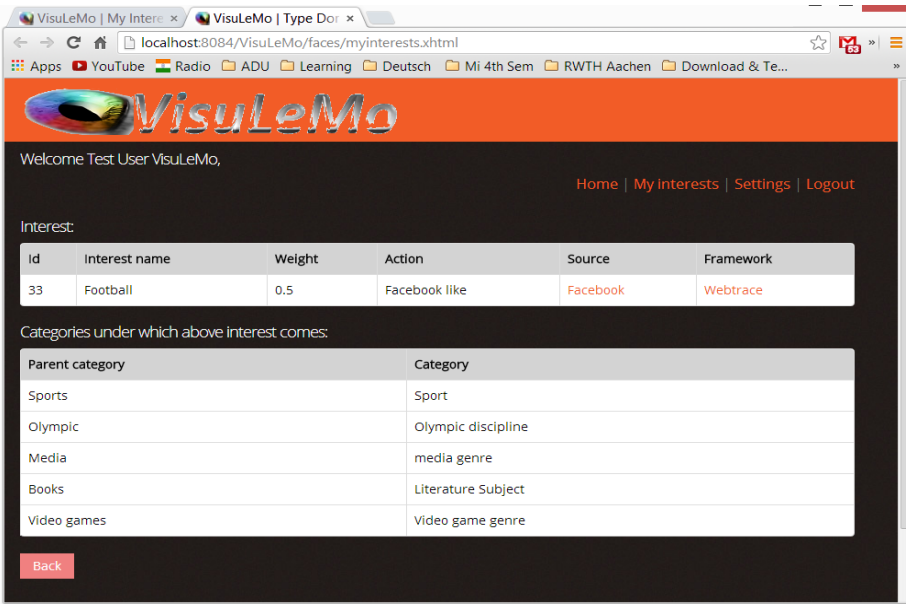
c) My Interests page.



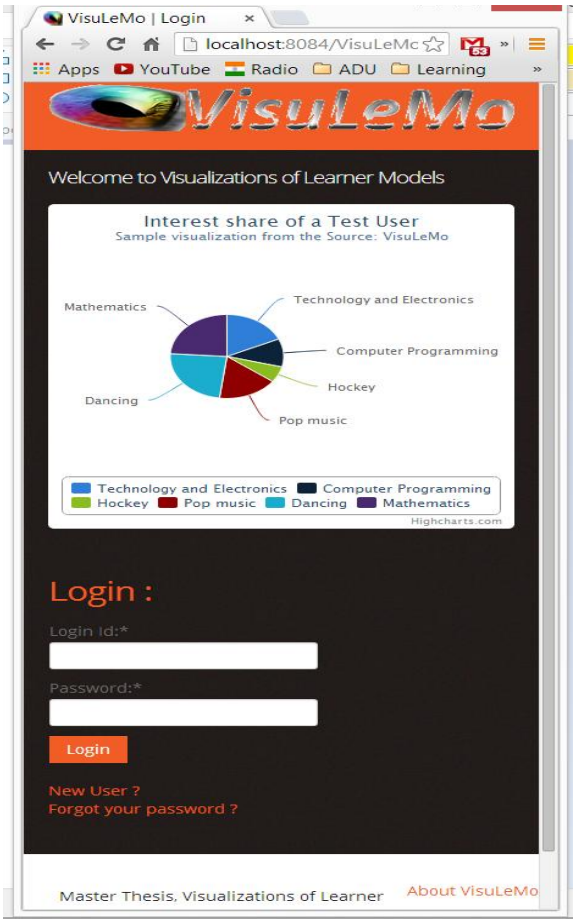
d) Visualization page.

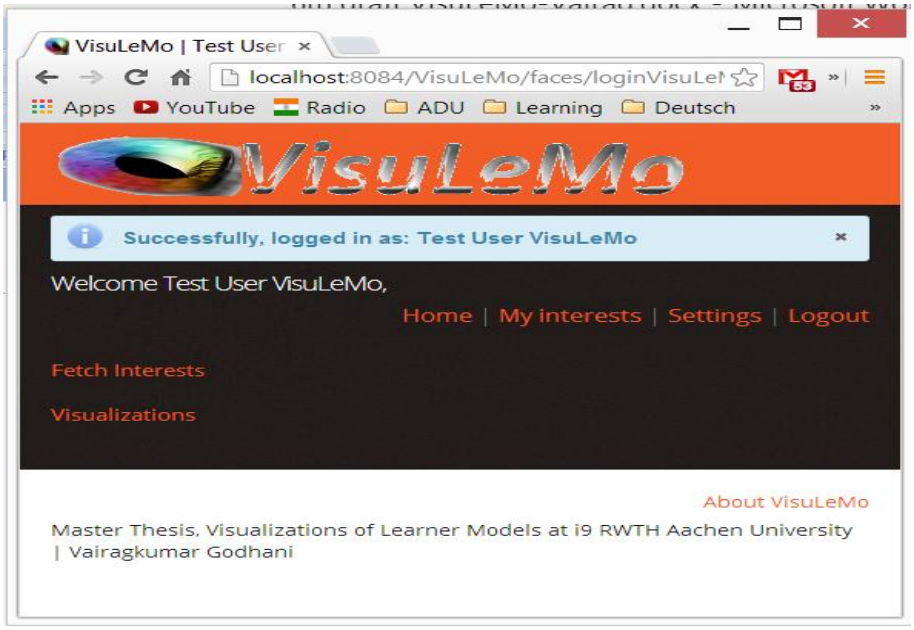


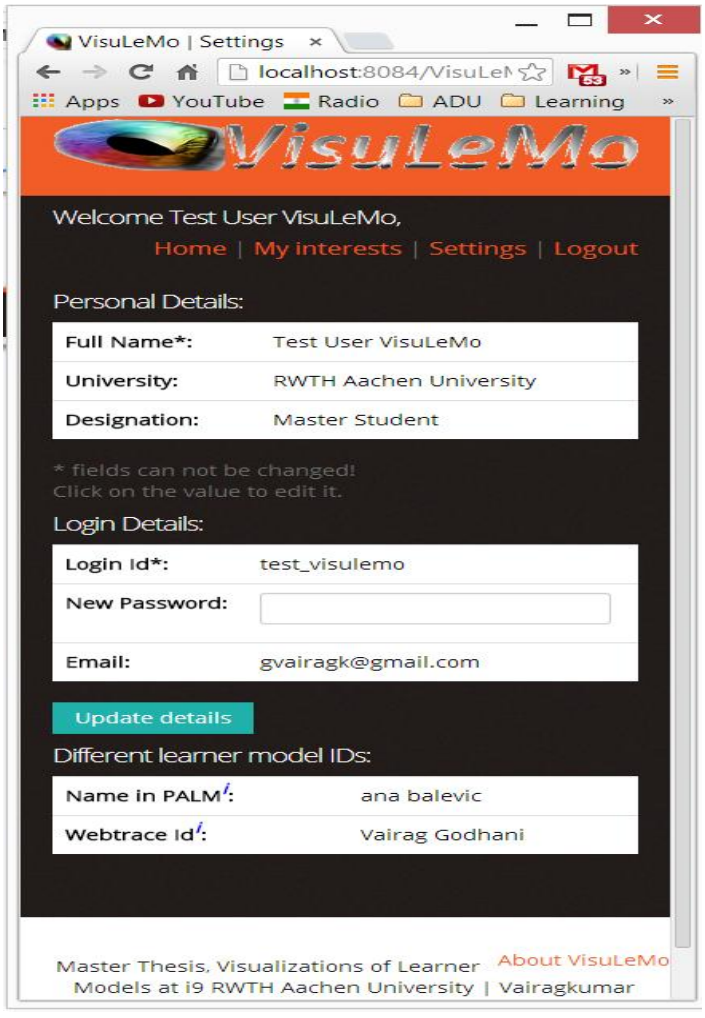
e) Page for viewing categories of an interest



f) Responsiveness of VisuLeMo pages (shown with different screen sizes).




VisuLeMo | Test User
localhost:8084/VisuLeMo/faces/loginVisuLeM
Apps
YouTube
Radio
ADU
Learning
Deutsch
VisuLeMo
Successfully, logged in as: Test User VisuLeMo
Welcome Test User VisuLeMo,
Home | My interests | Settings | Logout
Fetch Interests
Visualizations
About VisuLeMo
Master Thesis, Visualizations of Learner Models at i9 RWTH Aachen University | Vairagkumar Godhani


VisuLeMo | Settings
localhost:8084/VisuLeM
Apps
YouTube
Radio
ADU
Learning
VisuLeMo
Welcome Test User VisuLeMo,
Home | My interests | Settings | Logout
Personal Details:
Full Name*: Test User VisuLeMo
University: RWTH Aachen University
Designation: Master Student
* fields can not be changed!
Click on the value to edit it.
Login Details:
Login Id*: test_visulemo
New Password:
Email: gvairagk@gmail.com
Update details
Different learner model IDs:
Name in PALM: ana balevic
Webtrace Id: Vairag Godhani
Master Thesis, Visualizations of Learner Models at i9 RWTH Aachen University | Vairagkumar
About VisuLeMo