DISSERTATION / DOCTORAL THESIS

Titel der Dissertation / Title of the Doctoral Thesis

“An Intelligent Integrated Computer-Assisted Language Learning (iiCALL) Environment”

verfasst von / submitted by

Dipl.-Ing. Harald Wahl

angestrebter akademischer Grad / in partial fulfilment of the requirements for the degree of

Doktor der technischen Wissenschaften (Dr. techn.)

Wien, Februar 2017 / Vienna, February 2017

Studienkennzahl lt. Studienblatt / degree programme code as it appears on the student record sheet:	A 786 881
Dissertationsgebiet lt. Studienblatt / field of study as it appears on the student record sheet:	Informatik (Stzw)
Betreut von / Supervisor:	ao. Univ.-Prof. Mag. Mag. Dr. Werner Winiwarter

Acknowledgments

First of all, many thanks to ao. Univ.-Prof. Mag. Mag. Dr. Werner Winiwarter for his continuous support. Without his initial suggestion to choose a topic in the area of language learning and without his inspiring assistance, this thesis would have never been realized. Furthermore, I would like to thank all those students at the University of Applied Sciences Technikum Wien who were interested in the topic and contributed pieces of work within their master theses, bachelor theses, or practical works within their course programs: primarily, Rudolf Galler with his software development skills, and additionally, Mario Bucsics, Mira Dragasevic-Sperling, Werner Frühwirth, Kamil Havlicek, Nikolaus Kaiser, Auguste Mandl, Christian Nusser, Michael Taus, Camelia Tegrovsky, and Manfred Zangl. Finally, thanks to all who understood that I sometimes made myself rather scarce.

Abstract

In this thesis we present a software framework, a generic data model, and a development method for implementing an integrated language learning platform.

Computer-Assisted Language Learning (CALL) has a long tradition. It already started in the 1960s trying to support teachers and learners in language learning. Through innovations in the areas of Artificial Intelligence and Computer Linguistics, CALL systems have developed to Intelligent Computer-Assisted Language Learning (ICALL) systems. New progress in Natural Language Processing (NLP), which has become an increasingly active research discipline, enabled several applications in the field.

For a language learning platform being a state-of-the-art software product, we identify the requirements of extensibility, flexibility, and re-usability. Within the variety of existing language learning platforms, we perceive a lack of those requirements. This leads to the necessity to have a proper development process that covers functional requirements engineering as well as technical software design.

The work on this thesis emphasizes context-related learning. Context-relatedness is meant as improving skills within specific life situations. We concentrate on creating an integrated natural language e-learning system to be used within common working environments. Such environments, like, for instance, Web browsers or email clients, use a specific content as basis for language learning scenarios. In our research, we see "integration" from a broader perspective including also the whole development process. Our development process starts from requirements analysis and ends with technology-oriented specifications for implementations. We call ICALL systems being integrated in such a way Intelligent Integrated Computer-Assisted Language Learning (iiCALL) systems.

In this thesis we introduce a system architecture, a generic and extendable data model, as well as a structured way to design learning scenarios. As a prerequisite, we evaluate NLP tools and toolkits as candidates for being integrated in the system. For the design of iiCALL systems, we compare technical ideas from the healthcare domain to the language learning domain. As a result, we adapt the Health Level Seven International (HL7) V3 standard with its Reference Information Model (RIM) and corresponding Healthcare Development Framework (HDF) to iiCALL. We highlight the evolution of prototypes, starting from an intuitive system architecture to the Version 1 of the iiCALL Development Framework. To show the feasibility of our iiCALL research, we present the prototype of an automatic exercise learning scenario which is based on our iiCALL Software Framework and the iiCALL Generic Data Model.

Kurzfassung

Diese Arbeit präsentiert ein Softwareframework, ein generisches Datenmodell sowie ein Vorgehensmodell zur Entwicklung einer integrierten Sprachlernplattform.

Computerunterstütztes Sprachenlernen hat eine langjährige Entwicklung hinter sich. Schon in den 1960er Jahren waren unter der Bezeichnung „Computer-Assisted Language Learning (CALL)“ erste Implementierungen verfügbar. In den darauf folgenden Jahren wurden die Forschungsergebnisse aus den Bereichen der Künstlichen Intelligenz und der Computerlinguistik in die Sprachlernumgebungen integriert, wofür sich die Bezeichnung „Intelligent Computer-Assisted Language Learning (ICALL)“ etabliert hat.

Derzeit verfügbare Sprachlernplattformen genügen nur bedingt den Erfordernissen moderner Softwaresysteme. Erweiterbarkeit, Flexibilität und Wiederverwendbarkeit sind Anforderungen, welche man durch einen standardisierten Entwicklungsprozess erreichen kann, der sowohl Anforderungsanalyse als auch technische Implementierungsvorgaben beinhaltet.

Diese Arbeit konzentriert sich auf kontextbezogenes Sprachenlernen. Das bedeutet, dass Lernfortschritt situationsbezogen möglich sein soll. Ziel der Arbeit ist die Entwicklung eines E-learning Systems, das in verschiedene Arbeitsumgebungen wie etwa in Webbrowser oder Emailprogramme integriert werden kann und dort den aktuell betrachteten Inhalt als Grundlage für das Sprachenlernen nutzt. Wir betrachten die „Integration“ von einer breiteren Perspektive aus und meinen damit zusätzlich auch den kompletten Entwicklungsprozess, der, ausgehend von einer Anforderungsanalyse, technische Spezifikationen für eine Realisierung ermöglicht. Wir bezeichnen ein solches System als „Intelligent Integrated Computer-Assisted Language Learning (iiCALL)“ System.

In dieser Arbeit wird eine Systemarchitektur und ein generisches und erweiterbares Datenmodell vorgestellt, sowie ein klar strukturiertes Vorgehensmodell zur Implementierung von Use Cases für Sprachenlernen entwickelt. Dafür werden bestehende Natural Language Processing (NLP) Tools nach ihrer Eignung zur Integration in ein iiCALL System evaluiert. Der Ansatz, Entwicklungen aus anderen Domänen (im Speziellen aus dem Gesundheitswesen) für die Verwendung im Sprachenlernen adaptieren zu können, führt zum Health Level Seven International (HL7) V3 Standard und dem dort verwendeten Reference Information Model (RIM) sowie dem dazugehörigen Healthcare Development Framework (HDF). Die Arbeit beschreibt die Architekturentwicklungen, das iiCALL Development Framework und einen aktuellen Prototypen für einen Use Case zur automatischen Erstellung von Sprachtests, der technisch auf dem entwickelten iiCALL Software Framework und dem iiCALL Generic Data Model beruht.

Contents

Part I

Introduction and Backgrounds

CHAPTER 1

Introduction

Natural Language Processing (NLP) is a sub-discipline of computer science, which has developed technologies for several application cases like machine translation, automatic content extraction, speech recognition, Information Retrieval (IR), etc. (for some fundamentals of NLP see Chapter 4). This thesis deals with Computer-Assisted Language Learning (CALL), which emerged in the 1960s. Specific CALL systems are called Intelligent Computer-Assisted Language Learning (ICALL) systems if they use Artificial Intelligence and NLP technologies. To be precise, we deal with ICALL systems.

1.1. Motivation and Context

Different online platforms offer varying language learning tasks. These platforms mostly provide structured courses or learning options mainly for everyday speaking skills. In this thesis, our research concentrates on ICALL to support context-related learning. Including a context in the learning process allows to focus on different life and business situations. Learners are enabled to acquire language skills in varying fields of application, be it health, information technologies, business, etc. Additionally, we focus on integrated language learning that allows learning within common working environments like Web browsers or email clients.

A state-of-the-art language learning platform must be a software product that fulfills requirements of extensibility, flexibility, and re-usability. Within the range of existing language learning platforms, we perceive a lack of those requirements. To fulfill those requirements, we see the necessity to define a proper development process that covers functional requirements engineering as well as technical software design. In our research, we define "integration" from a broader perspective including also such a development process. Only if this condition is satisfied, we call an ICALL system an Intelligent Integrated Computer-Assisted Language Learning (iiCALL) system.

1.2. Research Questions and Hypotheses

From a technical point of view, iiCALL must, by default, be a flexible, extensible and interoperable language learning system supporting open collaboration. There does not exist a comparable language learning system, so for developing the Intelligent Integrated Computer-Assisted Language Learning environment, in this thesis we answer the following research questions and corresponding hypotheses.

Research Questions.

Research Question 1. *How can a system architecture, an underlying data model, and a technology-oriented development process look like to fulfill the requirements of an Intelligent Integrated Computer-Assisted Language Learning environment?*

Research Question 2. *Is there a comparable model that can be adapted to create the Intelligent Integrated Computer-Assisted Language Learning system?*

Hypotheses.

Hypothesis 1. *An extendable software framework, which can integrate Natural Language Processing functionalities, and which is based on a generic data model allows the implementation of an Intelligent Integrated Computer-Assisted Language Learning system.*

Hypothesis 2. *Due to similarities to the health domain, the ideas of the Health Level Seven International can be adapted for the Intelligent Integrated Computer-Assisted Language Learning development.*

1.3. Contributions

Our contributions in this thesis can be subdivided into contributions to Language Learning Software Development and contributions to Academic Education. On the one hand, we introduce a development process that allows to implement language learning scenarios based on requirements analysis. On the other hand, the created language learning software can assist academic education in the area of language learning and in the area of Software Engineering, respectively.

Our approach and the progress of the development and implementation of Intelligent Integrated Computer-Assisted Language Learning has continually been published and discussed at several conferences. For a complete list of our publications see Chapter 2.

Contributions to Language Learning Software Development. Technically, functionalities of the iiCALL platform are based on NLP technologies and NLP resources enriched with semantic data. We provide a comprehensive comparison of NLP tools and toolkits from the perspective of being suitable candidates for inclusion in an iiCALL environment.

An iiCALL system is an e-learning platform supporting different types of learners. Corresponding learning theories are presented. Due to a large variety of learning methods, the huge amount of possible learning scenarios, high complexity and diversity of natural languages, and an immense number of imaginable supporting technologies and systems, the proposed iiCALL system needs to support flexibility, extensibility, and re-usability as default prerequisites.

In this thesis, a suitable framework architecture and an underlying generic data model which satisfies the specified requirements are developed. Several approaches of learning software architectures are compared. We present our solution as an adaption of technologies and models from the healthcare domain. The iiCALL generic data model is based on the HL7 Reference Information Model. The development process of iiCALL is based on the HL7 Healthcare Development Framework. The development process starts from requirements analysis and ends with technology-oriented specifications for implementations. Following the iiCALL evolution, we illustrate corresponding implementations and the development of prototypes.

The evaluation at the end of thesis proves that our iiCALL solution fulfills the specified requirements for the iiCALL approach.

Contributions to Academic Education. The design of the iiCALL development process allows different kinds of users to participate in the creation of new language learning scenarios. Language teachers and educators may design new exercises, linguists and computer linguists may define new linguistic tools, and developers get clear specifications for implementation purposes.

Besides contributions to language learning, our research also accounts for academic education in the field of Software Engineering. The development of the iiCALL system is continuously integrated in courses at the University of Applied Sciences Technikum Wien. Based on the idea of research-based teaching, we demonstrate how the iiCALL environment can support teaching in the field of software programming or software project management.

1.4. Organization of Thesis

This thesis consists of four parts. In the first Part "Introduction and Backgrounds" we describe the motivation for our iiCALL research and indicate our contributions to Language Learning Software Development and to Academic Education (see Chapter 1). In Chapter 2 we list all our publications related to iiCALL research, and in Chapter 3 we give a comprehensive overview of related work and activities.

In the second Part "Towards the Intelligent Integrated Computer-Assisted Language Learning Environment" we present relevant NLP fundamentals (see Chapter 4), and in Chapter 5 we compare NLP toolkits that can suitably be integrated in the iiCALL environment. In Chapter 6 we compare the healthcare domain with the language learning domain and evaluate the HL7 standard to be adapted for developing the iiCALL system.

In the third Part "Implementation" we describe the evolution of the iiCALL development. Chapter 7 shows the first iiCALL architecture and the first prototype (a context-related vocabulary trainer). In Chapter 8 we introduce the iiCALL Development Framework, which is based on the HL7 V3 standard. Requirements for the core iiCALL system are addressed in Chapter 9. In Chapter 10 we explain the development process of the "Automatic Exercise Learning Scenario" based on the phases of the iiCALL Development Framework, and in Chapter 11 we present the corresponding implementation details.

Finally, in the Part "Results and Prospects" we discuss and evaluate the results of our iiCALL research. In Chapter 12 we evaluate our iiCALL research results with respect to the requirements of the iiCALL environment, and answer our Research Questions and Hypotheses. In Chapter 13 we explain the educational impacts of the iiCALL research, and in Chapter 14 we summarize our contributions and refer to future research.

CHAPTER 2

Dissemination

Since 2010, we have continually published and discussed the idea and the progress of developing and implementing our Intelligent Integrated Computer-Assisted Language Learning research at several conferences. At the 12th International Conference on Information Integration and Web-based Applications & Services (iiWAS 2010) (see [**Wahl et al., 2010**]), we compared and evaluated NLP tools to fulfill the needs of a planned language learning environment.

An extended version of this paper was selected for publication in the International Journal of Pervasive Computing and Communications (see [**Wahl et al., 2011**]), where we additionally showed a first draft of a possible system architecture. At the World Conference on Educational Multimedia (EdMedia) 2011 (see [**Wahl and Winiwarter, 2011a**]), our system architecture became more precise, and we specified learning methods for different levels of learners correlated to learning theories. At the 13th International Conference on Information Integration and Web-based Applications (iiWAS 2011) (see [**Wahl and Winiwarter, 2011b**]), we added exemplary use cases.

In 2012, at the 11th International Conference on Web-based Learning (ICWL 2012) (see [**Wahl and Winiwarter, 2012b**]), and at the 14th International Conference on Information Integration and Web-based Applications (iiWAS 2012) (see [**Wahl and Winiwarter, 2012a**]), we presented a Firefox plug-in providing a context-related vocabulary trainer.

A year later, in the paper at the World Conference on Educational Multimedia (EdMedia) 2013 (see [**Wahl and Winiwarter, 2013a**]), we discussed educational impacts of iiCALL. So far, the architecture was pretty inflexible with respect to new functionality due to hard-coded process steps and fixed data models at client and server side. The idea of a generic data model based on the HL7 Version 3 Reference Information Model, we firstly showed at the 15th International Conference on Information Integration and Web-based Applications (iiWAS 2013) [**Wahl and Winiwarter, 2013b**].

We concretized our idea in 2014 at the 16th International Conference on Information Integration and Web-based Applications (iiWAS 2014) (see [**Wahl and Winiwarter, 2014**]) by the introduction of the iiCALL Development Process and the mapping of use cases to the iiCALL Refined Message Information Model.

In 2015, we introduced the current iiCALL Software Framework based on the iiCALL Generic Data Model (GDM) with a corresponding prototype at the 14th International Conference on Web-based Learning (ICWL 2015) (see [**Wahl et al., 2015a**]), and the 17th International Conference on Information Integration and Web-based Applications (iiWAS 2015) (see [**Wahl et al., 2015b**]).

Finally, in the paper at the 18^{th} International Conference on Information Integration and Web-based Applications (iiWAS 2016), we dealt with extending functionality using the Open Source iiCALL Software Framework (see [**Wahl et al., 2016**]).

Three master theses at the University of Applied Sciences Technikum Wien, supervised by the author of this thesis, dealt with iiCALL-related topics. In 2013, Michael Taus discussed language and context recognition (see [**Taus, 2013**]). In 2014, Kamil Havlicek worked on Web browser plug-in development as possible client platform for iiCALL (see [**Havlicek, 2014**]). Finally, Rudolf Galler worked on the framework implementation based on the iiCALL Generic Data Model in 2015 (see [**Galler, 2015**]).

CHAPTER 3

Related Work and Activities

In the area of language learning and NLP, several different research activities are currently in progress. Our research is located in the context of CALL using NLP methods. Historically, research on using NLP technologies in CALL began in the 1980s and has continuously evolved. For appropriate overviews see [**Antoniadis et al., 2013, Greene et al., 2004, Callmeier et al., 2004**]

Several interest groups deal with NLP with a specific focus on language learning, for example, the Association for Computational Linguistics (ACL)[1], the European Association for Computer-Assisted Language Learning (EUROCALL)[2], the International Speech and Communication Association (ISCA)[3], or the Computer-Assisted Language Instruction Consortium (CALICO)[4].

On the level of research in the European Union (EU), in the Framework Programme 7 (FP7), the ICT "Challenge 4: Technologies for Digital Content and Languages" and "Challenge 8: ICT for Learning and Access to Cultural Resources" came with several calls in the field. In the new Horizon 2020 (H2020) Research and Innovation Programme, again some topics are in the same context, e.g. "ICT-20-2015: Technologies for better human learning and teaching" or "ICT-22-2016: Technologies for Learning and Skills".

3.1. Existing Language Learning Platforms

Existing language learning platforms like Anki, Bab.la, Babbel, Babelyou, Busuu, Dalango, Memomo, Wijn, Quizlet, WizIQ, etc. offer exercises, tutorials, videos, chats and communication tasks. They follow several teaching and learning strategies, and, therefore, they provide a variety of supporting tools. However, all of these platforms are not "integrated", meaning that they cannot be used within electronic working environments like Web browsers or email clients in a way that they use the actual content of the environment such as a currently opened Web site or email.

Anki. The language learning program Anki[5] offers learning based on the principle of flash cards. Flash cards aim to ease remembering things. The two main concepts of Anki are spaced repetition (first published in 1972 [**Craik and Lockhart, 1972**]) and active recall testing. Spaced repetition is a specific learning method that follows the idea of repeating learning

[1]`https://www.aclweb.org/`
[2]`http://www.eurocall-languages.org`
[3]`http://www.isca-speech.org`
[4]`http://www.calico.org`
[5]`http://ankisrs.net`

matter within changing time intervals. Active recall testing is a technique of asking questions and remembering linked answers. Anki provides downloads of installation software for several platforms, and is also available for mobile devices. An online version[6] additionally offers the option to synchronize personal flash cards. For developers, even the source code for Linux/BSD operating systems is available.

bab.la. The project babLa[7] offers an online learning platform which provides its Web site in 28 different languages. It is free of charge and includes lots of functionalities like bilingual dictionaries, learning quizzes and games, phrase books, context-related sentences, and vocabulary training. A dictionary browser plug-in for Chrome and Firefox for finding translations, and some Web widgets are available for download.

Babbel. Babbel[8] is an online learning platform, which is also available for smartphones and tablets. It offers services for two levels of learners: beginners or advanced. The learning focus is on real life situations which are presented by several interactive training lessons using multimedia.

Babelyou. The free online learning site Babelyou[9] offers videos and exercises for many different languages. Additionally, it allows to find other people for exchanging personal language skills. The main idea is that one can make further progress by communicating with real users, so-called exchange partners or eTandems.

Busuu.com. Busuu[10] comes with both, a free and a commercial version. The free online version simply allows to work with vocabulary cards and exercises for text writing. Even corrections done by native speakers are possible. The commercial version (also available as mobile app with an offline option) additionally offers specific courses for traveling or grammar exercises. Offered learning courses are correlated to the Common European Reference Framework for Languages (CEFR)[11] according to the levels A1, A2, B1, and B2. Premium users can also obtain an official certificate from a known partner institute.

Dalango. Dalango[12] is the commercial Web version of language learning provided by the "Spotlight Verlag", which is a German publishing company for printed language learning magazines. The online version is only available in German, and it provides learning of English (and also Business English), French, Spanish, and Italian. Used learning methods are supported by videos and correlated exercises for grammar and vocabulary issues in the context of communication, daily routines, cooking, health, hotels, nightlife, or traveling. Intentionally, videos show people with accent to simulate realistic real life environments. With a specific focus on Business English, the

[6]`http://ankiweb.net/`
[7]`http://bab.la`
[8]`http://www.babbel.com`
[9]`http://www.babelyou.com`
[10]`http://www.busuu.com`
[11]`http://www.coe.int/t/dg4/linguistic/Source/Framework_EN.pdf`
[12]`http://www.dalango.de/`

platform tries to highlight working life scenarios realized by videos of job interviews, presentations, telephone calls, or negotiations.

Memomo. Memomo[13] is a Web site that offers vocabulary training using the so-called learning box methodology. It supports the languages English, German, French, Spanish, and Italian. For quality progress, users can report wrong translations in the system's database.

Quizlet. Quizlet[14] offers an interactive vocabulary learning scenario based on flashcards. When learning, users can interrupt at any time to continue later. Additionally, users can customize their individual learning sets. The providers of Quizlet indicate that the functions of the Web site are perfectly suited to be used for every curriculum, at every level and also by every teacher or student worldwide.

WizIQ. WizIQ[15] offers virtual classes, online tests, and educational content for learners. Teachers can use and administrate different virtual classrooms or integrate functionalities into the Learning Management Platform Moodle[16] by Web conferencing modules.

Summary. The above mentioned learning platforms offer language learning using varying learning methods. They do not use NLP technologies and therefore, they are CALL systems but not ICALL systems. Besides, they offer no information or possibilities to extend them by implementing individual source code. They are meant to be used "as-is".

3.2. Existing Integrated Language Learning Platforms

There also exist some learning platforms that are integrated in working environments. Such platforms like Language Immersion, Mind the Word, Ming-A-Ling (not available any more), Polyglot, or Visual Input Enhancement of the Web (VIEW) are mostly Web browser plug-ins providing a variety of different learning tasks.

Language Immersion. The development of Language Immersion[17] was started in 2012 by an American Company called Use All Five[18]. The experimental Google Chrome Web browser extension Language Immersion supports the main functionality of automatic language translation applicable to a huge variety of natural languages. It uses the Google Translate API to translate words or phrases on a Web site. This way, they promise learners to get a deeper knowledge of language usage in a specific context.

[13]`http://memomo.net/`

[14]`https://quizlet.com`

[15]`https://www.wiziq.com`

[16]`https://moodle.org/`

[17]`https://chrome.google.com/webstore/detail/language-immersion-for-ch/bedbecnakfcpmkpddjfnfihogkaggkhl`

[18]`http://www.useallfive.com`

Mind the Word. Mind the Word[19] is an Open Source Google Chrome extension released by OiWorld[20]. At the Github Web site, where the source code can be downloaded, it promises to be "an extension for Google Chrome that helps people learn new languages while they browse the web"[21]. Therefore, the software chooses a few words of a currently viewed Web site and translates them into a language the user wants to learn. One can configure the preferred target language and the percentage of text that shall be translated. Personal dictionaries and translations can be stored for further look-up.

Ming-A-Ling. The Firefox add-on Ming-A-Ling[22] was initially released in 2010. While browsing the Web, a learner can add words he is familiar with to a personal language library. Words in the library will be translated automatically into the target language a user wants to learn. Since Google started to restrict the usage of the Google Translate API, the author has stopped further development and removed the add-on from Mozilla's add-on directory.

Polyglot. Polyglot[23] is another Google Chrome add-on that translates randomly words from a site a user visits. As Ming-A-Ling, Polyglot is no longer maintained due to the limitations of the Google Translate API.

Visual Input Enhancement of the Web (VIEW). The Intelligent Computer-Assisted Language Learning system Visual Input Enhancement of the Web (VIEW)[24] offers several integrated language learning features within the Web browsers Firefox, Chrome, and Opera. It evolved as a multilingual extension of the Working with English Real Texts (WERTi)[25] system, which was originally developed at Ohio State University in 2006. Through NLP technologies like tokenization, lemmatization, or morphological analysis, VIEW provides different exercises like multiple choice tests or cloze tests to the users.

Summary. All above mentioned integrated tools with the exception of VIEW offer only limited features for language learning. Some are even no longer supported due to license restrictions of the Google Translate API. However, VIEW is a very useful and comprehensive tool, the only one that can be truly called an ICALL system, which makes intense use of NLP technologies applied to language learning methods [**Meurers, 2013**]. VIEW does not provide any information about the underlying software architecture. In [**Meurers et al., 2010**], a kind of a system architecture of VIEW is given, but it shows more or less only process steps and corresponding data flows of the mentioned client-server architecture. It seems that the research

[19]`https://chrome.google.com/webstore/detail/mind-the-word/fabjlaokbhaoehejcoblhahcekmogbom`

[20]`http://www.oiworld.org`

[21]`https://github.com/OiWorld/MindTheWord`

[22]`https://addons.mozilla.org/en-us/firefox/addon/ming-a-ling/`

[23]`https://chrome.google.com/webstore/detail/polyglot/plpjkjplknknmhfhkjgcfgofclmlnine`

[24]`http://sifnos.sfs.uni-tuebingen.de/VIEW/`

[25]`http://sifnos.sfs.uni-tuebingen.de/WERTi/`

focus of the VIEW project is more on the development of ICALL services than on a universal and extensible software architecture.

3.3. Learning Theories and Their Impact on Learning Methods

Learning theories and methods were already discussed long before computers were used for learning, even, long before computers were invented. Behaviorism, Cognitivism, and Constructivism are such learning theories. Merging the ideas behind those learning theories with the variety of computer-assisted learning methods is the basis for specific learning scenarios for the iiCALL system [**Wahl and Winiwarter, 2011a**]. Amaral and Meurers discuss important additional facts to keep in mind. An ICALL system can be useful in language learning if, on the one hand, the system addresses the learners' needs and, on the other hand, pedagogical issues are part of the activity design [**Amaral and Meurers, 2011**].

Behaviorism. Behaviorism follows the idea that learning can be achieved by a change of behavior. The most famous pioneers and thinkers of Behaviorism are Ivan P. Pavlov (1849-1936), John B. Watson (1878-1958), and Burrhus F. Skinner (1904-1990). Pavlov, a Russian psychologist and physician, is well known for his conditioning experiments of dogs. During the experiment, he repeatedly rang a bell and, right after ringing, served meat to the dogs. The dogs' natural (unconditioned) reaction to the presentation of the meat was an increased production of saliva. After several iterations, the experiment changed in a way that after the bell ringing no more meat was presented to the dogs. Pavlov could reproducibly observe the dogs' (now conditioned) reaction that they did again produce increased saliva although no meat was offered. Comparably, Watson, an American psychologist, started his experiments with animals, and later on, he worked on conditioning of human beings. A well known sentence by him showing strikingly the success of his research:

> "Give me a dozen healthy infants, well-formed, and my own specified world to bring them up in and I'll guarantee to take any one at random and train him to become any type of specialist I might select doctor, lawyer, artist, merchant-chief and, yes, even beggar-man and thief, regardless of his talents, penchants, tendencies, abilities, vocations, and race of his ancestors. I am going beyond my facts and I admit it, but so have the advocates of the contrary and they have been doing it for many thousands of years." [**Watson, 1930**]

Skinner, also an American psychologist, developed a method in the 1950s, which he called "programmed instruction – drill and practice" based on operant conditioning. In contrast to classical conditioning, which deals with reflexive behaviors, in operant conditioning change of behavior is achieved by evaluating consequences through positive and negative reinforcement [**Domjan, 2009**]. An overview of the principles of operant behavior is shown in Figure 1.

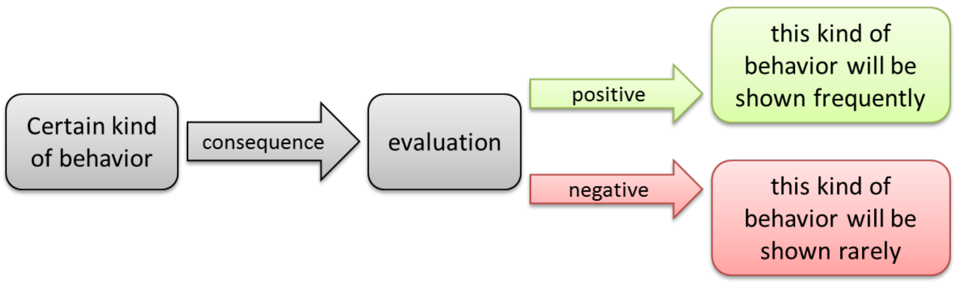

Certain kind of behavior
consequence
evaluation
positive
this kind of behavior will be shown frequently
negative
this kind of behavior will be shown rarely

FIGURE 1. Principles of Operant Behavior

Behaviorism sees mind as a kind of "black box", it does not observe the mind. It simply considers learning as a change of behavior generated by a combination of stimulus and response. Typical processes based on the principles of Behaviorism are more or less linear, e.g.

(1) knowledge module 1
(2) test question 1
(3) correct answer: knowledge module 2
(4) incorrect answer: knowledge module 1 again
(5) test question . . .
(6) . . .

Such kinds of learning processes work well, e.g. for grammar lessons or vocabulary trainings. Immediate feedback can be provided, errors can be corrected directly. Incorrect answers force to re-do the lesson, correct answers yield to further learning steps. An individual learning speed is possible and progress can easily be measured. Learning is predetermined by others, including questions and the corresponding set of correct answers. There is no option for freely formulated answers. Individual interests or learning directions cannot be supported. Therefore, the kind of learning processes are limited, critical reflection is not a designed part of learning and individual learning progress cannot be transferred to new areas [**Buzzetto-More, 2006, Watson, 1913**].

Cognitivism. In contrary to Behaviorism, Cognitivism opens the "black box". Not only observable behavior but thinking and mental processing becomes important. Learners do not only react on stimuli, furthermore, they act as reaction of rational thinking and conclusions. Cognitivism uses the metaphor of the mind comparable to a computer: information comes in, then it is being processed, and a specific outcome is created [**Semple, 2000, Domjan, 2009**]. Cognitivism focuses on mental activities that take place when people are learning. It concentrates on how information is obtained, how it is organized and stored. Learning is less focused on what learners do, it is primarily focused on the way how learners acquire new knowledge [**Ertmer and Newby, 1993**].

Constructivism. Constructivism is a learning theory that indicates human knowledge as a construction built through interaction and experimenting. Learners perform by experimenting and learning by doing. Creating personal cases and social interaction yield to individual learning issues. Learning environments that support Constructivism must allow case-based settings instead of several representations of reality. Predetermined learning steps switch to an environment that supports individual questions and answers, e.g. by chats, social learning in teams, or virtual classrooms [**Buzzetto-More, 2006**].

Cognitive Load Theory. Cognitive Load Theory (CLT) is a theory developed by John Sweller in 1988, that deals with mental effort needed in working memory while learning [**Sweller, 1988**].

Working Memory, Long-term Memory, Schemata. According to the work of [**Sweller et al., 1998**], human beings' cognitive architecture can be summarized in the following way. There is a limited working memory which can hold only about seven elements of information at the same time. This working memory deals with all conscious activities, it processes information, it organizes, compares, and constructs. There exists a limitation of two or three items that can simultaneously be processed. Working memory can be compared to consciousness. All other cognitive functions are not available until they are brought into working memory. Baddely divided working memory into a "visual spatial scratch pad" and a "phonological loop" [**Baddeley, 1992**]. Besides this working memory, there is an effectively unlimited long-term memory that can be used to store schemata having different degrees of automatism. A schema categorizes elements of information according to the way in which they will be used [**Chi Michelene T. H., 1981**]. One of the apparent functions of a schema is to provide a mechanism for knowledge organization and storage. Additionally, schemata aim to reduce working memory load, because the number of elements that can be processed in working memory is limited, but their size and complexity are not. Schema construction is processed in working memory, after extensive practicing, automatism in schema construction occurs and automation can bypass working memory. Automated schemata allow both: fluid performance on familiar aspects of tasks and by freeing working memory capacity performance on unfamiliar aspects.

Intrinsic, Extraneous, and Germane Cognitive Load. John Sweller distinguishes between three kinds of cognitive loads: Intrinsic Cognitive Load (ICL), Extraneous Cognitive Load (ECL), and Germane Cognitive Load (GCL). Intrinsic Cognitive Load is given by the learning tasks themselves. Therefore, is is fixed by the learning material which cannot be altered. For non-interacting learning units that have no need to reference other units, i.e. units one can learn isolatedly, the ICL is low. Tasks with low intrinsic load can be learned serially. There is no need to hold many other elements in the working memory. Vocabulary learning indicates a typical example with low ICL. On the other hand, grammar learning has a high intrinsic load. Here, tasks have to be processed simultaneously and learning units interact highly. Learners on an expert level have already constructed schemata of interacting units. From the point of view of working memory, such interacting elements inside the schemata are not seen as single elements any more. Moreover, the schema in total is seen as one single element [**Sweller et al., 1990, van Merriënboer and Sweller, 2005**].

On the contrary, Extraneous Cognitive Load is defined by the manner in which the material is presented. Thus, is is determined by the instructional design. It reflects the strain that is needed to spend on designed instructions. Often, instructions include several sources of information like combinations of referring texts and diagrams. To be able to understand such a text and diagram, it is necessary to mentally integrate them. Such a kind of mental integration enforce a high ECL [**Sweller et al., 1990, van Merriënboer and Sweller, 2005**]. It is not recommended to design learning issues that need a combination of high Intrinsic Cognitive Load and high Extraneous Cognitive Load. This combination may yield to considerably exceeded working memory (see Figure 2).

The construction of schemata is called Germane Cognitive Load. There are some instructional procedures that support the construction of schemata. For instance, asking questions about an exercise, or providing exercises that students have to complete, encourage students to learn by increasing the GCL. Bannert mentions that Germane Cognitive Load "occurs when free working capacity is used for deeper construction and automation of schemata" [**Bannert, 2002**]. For best learning progress, the Germane Cognitive Load is desired to be maximized.

For the design of learning environments, ideally, all types of cognitive loads shall be taken into consideration. Learning progress can be maximized if the whole capacity of working memory can be used. Intrinsic Cognitive Load is predetermined and therefore unavoidable, so Extraneous Cognitive Load should be optimized in a way that the resources for Germane Cognitive Load can be maximized. The following graphic (Figure 2) shows the optimal size of intrinsic, extraneous, and germane cognitive load in relation to the complexity of learning issues.

For the design of learning environments two questions are important:

- How can the Extraneous Cognitive Load be reduced?
- How can the Germane Cognitive Load be increased?

Multimedia Learning and Expertise Reversal Effect. Extraneous Cognitive Load, as defined above, can be decreased by changing the representation of learning materials. [**Mayer and Moreno, 2003**] explained in their work "Nine Ways to reduce Cognitive Load in Multimedia Learning" that multimedia learning can decrease extraneous load. They base their idea on the hypothesis that human beings own different systems, one to handle pictorial material, and one to handle verbal material. They call it Dual Channel Assumption (DCA). Figure 3 (after [**Mayer and Moreno, 2003**]) illustrates this cognitive theory of multimedia learning.

Words and pictures are stored and handled in two separate memory systems. Pictures can help to ease the processing of textual information but one has to keep in mind that, for example, words presented as texts come into working memory using the same sensory memory, the eyes. Thus, a written text in combination with pictures consumes presumably more mental resources than narrated text with pictures does. Moreover, some important

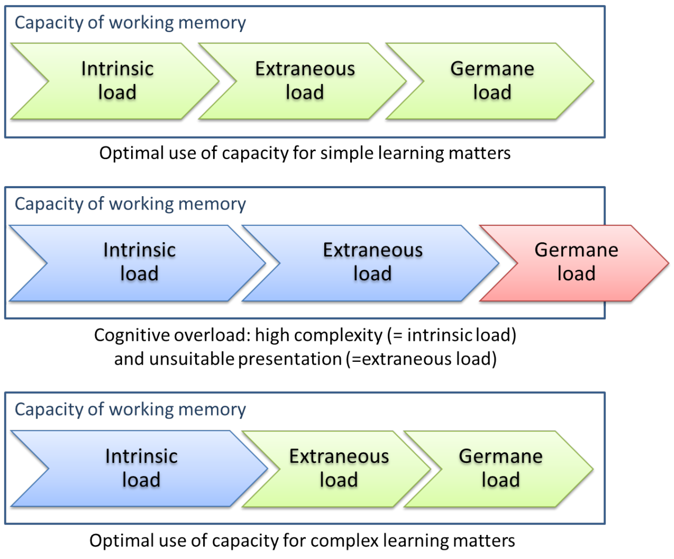

Capacity of working memory
Intrinsic load
Extraneous load
Germane load
Optimal use of capacity for simple learning matters
Capacity of working memory
Intrinsic load
Extraneous load
Germane load
Cognitive overload: high complexity (= intrinsic load) and unsuitable presentation (=extraneous load)
Capacity of working memory
Intrinsic load
Extraneous load
Germane load
Optimal use of capacity for complex learning matters

FIGURE 2. Use of Learning Memory Capacity

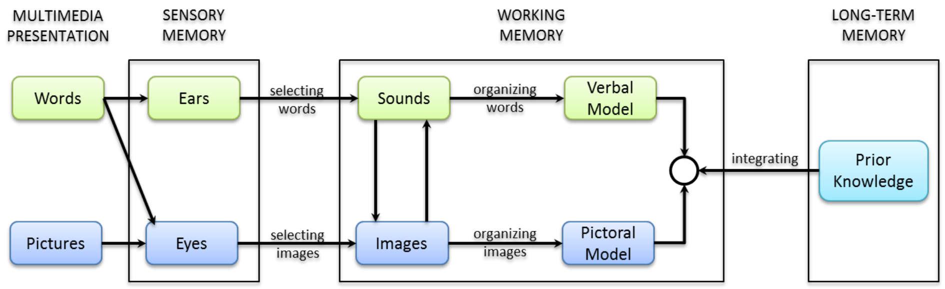


FIGURE 3. Cognitive Theory of Multimedia Learning

principles have to be considered in processing multimedia information for learning (see [**Triesman and Davies, 1973**]).

The principle of contiguity. Textual or vocal information should be presented concurrently or at least near in time and space with corresponding pictures. By deviation of this principle, a phenomenon of an exceeding increase of the Extraneous Cognitive Load can occur which leads to a worse learning performance. Chandler and Sweller described this impact and called it split attention effect [**Chandler and Sweller, 1992**].

The principle of coherence. Learning materials shall be organized in a way that learners can easily follow the thread. Human mind tends to relate new information with already stored information for understanding a specific context. A lack of coherence in a presentation of learning matters yields to an increase of extraneous load [**Mayer, 2009**].

The principle of redundancy. In learning matters, sometimes "less is more". It is more efficient to provide only necessary information focused on the learning essence than to distract students by redundant information. Decoration or background music, for instance, might goal a pleasant learning experience but contradicts targeted learning outcomes [**Mayer, 2009**].

The principle of individual learning. [**Kalyuga et al., 2003**] explained the Expertise Reversal Effect which means that the instructional techniques can be effective if the level of learner (with respect to the expertise) is taken into account. So, learning materials that are fully-guided may be highly effective for unskilled learners but may be inappropriate with more experienced learners.

Level of Learners. Appropriate learning scenarios best include prior knowledge and experiences of learners. Therefore, different levels need different learning styles and can be classified in the following way.

- *Beginner*: The prior knowledge of a beginner is low: a lack of necessary knowledge makes students to need help on a very low level which contradicts the fact that the learners do not understand learning materials. Often, assistance is seen as an additional requirement which is not manageable.
- *Intermediate*: The prior knowledge of an intermediate learner can be rated as medium. Learners who need help, have the ability to use provided help. Students have enough prior knowledge which they can build up for new information. Help is useful when it is offered within a specific reference frame.
- *Advanced, Expert*: The prior knowledge of an expert or an advanced learner is high. Learners have the capability to use given help, but there is no need for it. Explanatory learning material is mostly redundant.

In Table 1 (see [**Wahl and Winiwarter, 2011a**]), we give a selection of learning methods that fit the different levels of learners with regard to their representation in learning theories.

TABLE 1. Selection of Learning Methods Correlated to Learning Theories and Corresponding Level of Learners

Learning theory	**Behaviorism**	**Cognitivism**	**Constructivism**
Level of learner	Beginner	Intermediate	Advanced; Expert
Proper learning methods	Drill and practice; Linear structures; One correct answer possible; Vocabulary learning	Information processing; Dynamical structures; Multiple answers possible; Tutorial systems	Construction of information; Self-conducted learning; Social learning; Simulations

Summary. The presented theories play an important role in the design of the iiCALL environment. For instance, the learning theories Behaviorism, Cognitivism, and Constructivism have a direct relation to proper learning methods. Following the principle of individual learning, one can directly derive that differences in the levels of learners have to be taken into account.

Due to the principle of coherence, it is important to include context in the learning environment. Additionally, it forms the basis for the idea of integrating the learning action into working environments like Web browsers or email clients.

3.4. Standards in Natural Language Processing

In Information Technologies, standards have become more and more important. In the field of NLP, however, there are only a couple of standards already published. A sub-committee of the International Organization for Standardization (ISO)[26], for instance, focuses activities on Machine Readable Dictionary (MRD) and Lexical Resources (LR), and the interoperability issues between NLP programs and LR. ISO has published standards like the Lexical Markup Framework (LMF) and the Typed Feature Structure (TFS) Representation. Some other works have arisen in the field like different markup frameworks, e.g. SpatialML or TimeML [**Wahl et al., 2011**].

Lexical Markup Framework. The Lexical Markup Framework has been published as an ISO standard for NLP lexicons and MRD. It aims to present a general model for the use and the creation of LR. LMF handles data exchange between resources and the possibility to merge several resources to a common usable electronic resource [**Francopoulo et al., 2007**]. It comes with a meta-model to describe and represent lexical data that can be used within monolingual and multilingual software programs.

Revision 16 of the standard shows the core package which determines the fundamental hierarchy of a lexicon (see [**Francopoulo and George, 2008, ISO, 2008**]. Moreover, it shows enhancements to the core package by adding specifications that are useful for different lexical resources like MRD, syntactic structures, morphology of languages, semantic descriptions, etc.

Notations in LMF use the Unified Modeling Language (UML) and XML. The core package of the LMF is illustrated in Figure 4. A lexical resource is represented as single instance in the sense of singleton of design patterns. Therefore, a lexicon is a container for words in a specific language. A lexical entry allows the definition of connections between form and sense, etc.

The following Figure 5 shows the syntactic package of the LMF, and Figure 6 the semantic package of LMF. For the complete representation of LMF, we refer to [**ISO, 2008**].

Typed Feature Structure Representation. The Typed Feature Structure Representation is an option to define and describe grammatical information. For instance, some typical grammatical information is to describe a noun's number (singular or plural), the part of speech of a word (is the word used as noun, verb, etc.), or a verb's form (is it used as imperative, in which tense, etc.). Each feature structure consists of a set of features. TFS indicates a formalism to describe lexical grammar. The underlying model follows an object-oriented approach which also includes nested and even recursive declarations. Feature Structures are typically notated by matrices (a matrix example is shown in Figure 7) or by graphs (a graph example is shown in Figure 8 [**ISO, 2008**]).

[26]`http://www.iso.org/`

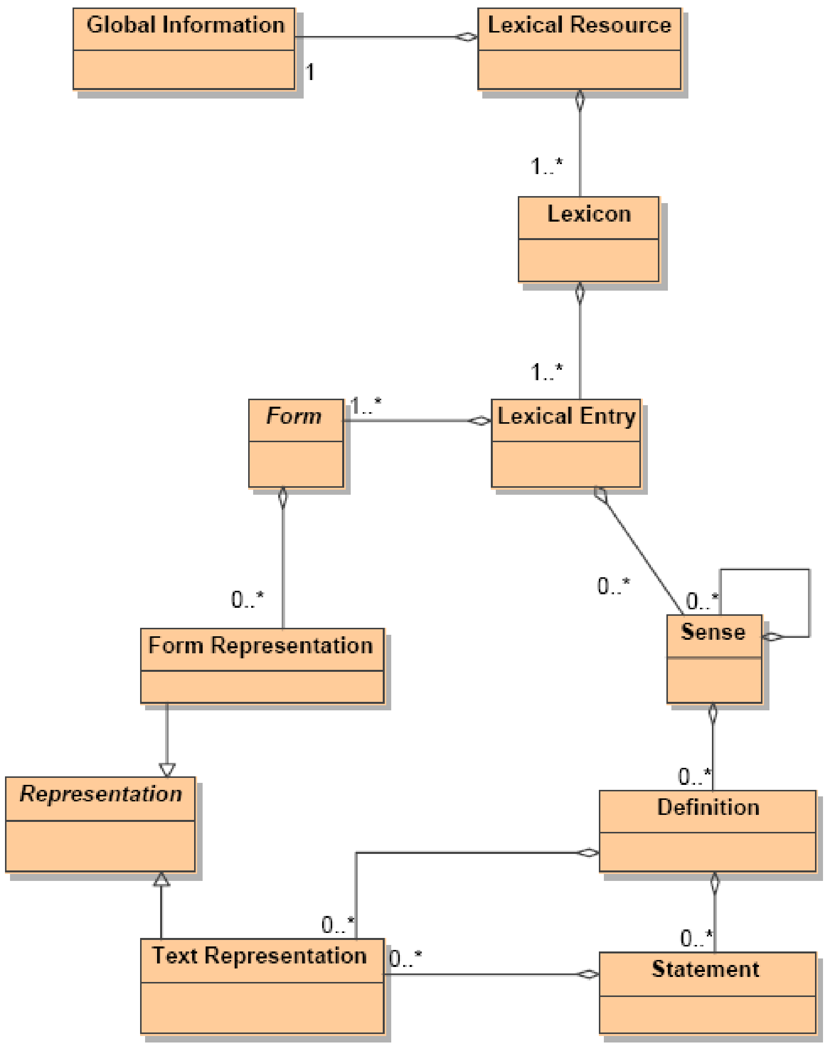


Figure 4. LMF Core Package [**ISO, 2008**]

Figure 9 shows a concrete example that notates the word "love" in matrix notation. Figure 10, respectively, illustrates it equivalently in a graph notation. It shows the orthography feature ("ORTH") and the syntax feature which again is split into two features: the part of speech ("POS", which has the value "verb"), and valence, which has the value "transitive" [**ISO, 2005b**]. This specific representation indicates that here the word "love" is used as a transitive verb [**Wahl et al., 2011**].

The TFS representation also provides notations in XML for easier processing by computer programs, the above example from Figure 9, for example, is described in XML as shown in Listing 1 [**Wahl et al., 2011**].

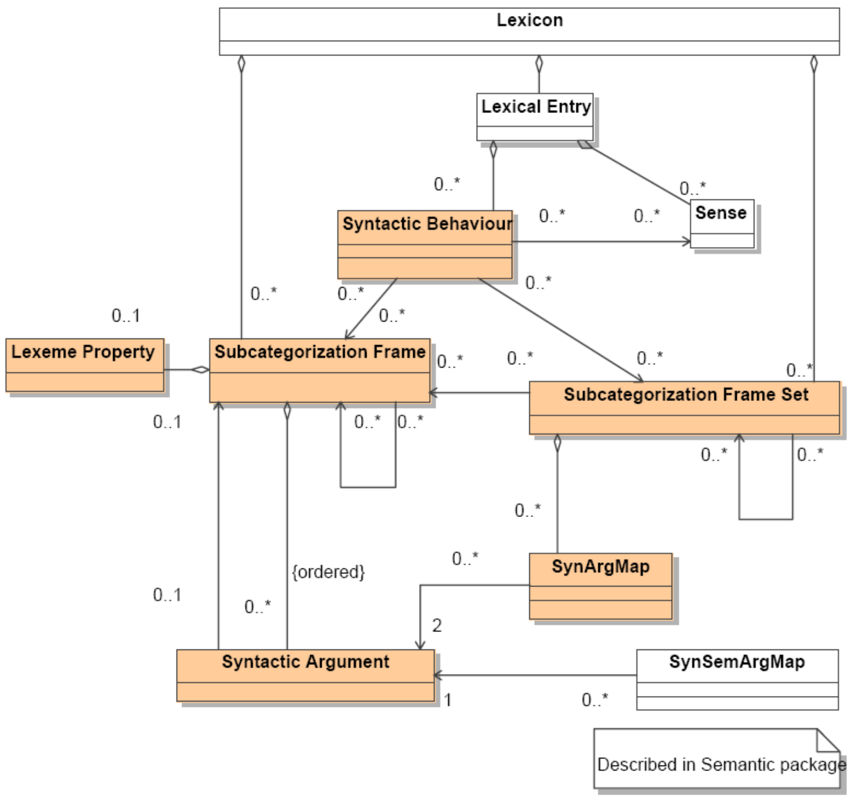


FIGURE 5. LMF Syntactic Package [**ISO, 2008**]

```
<fs>
  <f name="orth">
    <string>love</string>
  </f>
  <f name="syntax">
    <fs>
      <f name="pos">
        <symbol value="verb"/>
      </f>
      <f name="valence">
        <symbol value="transitive"/>
      </f>
    </fs>
  </f>
</fs>
```

LISTING 1. TFS Example

Additional Activities. In the field of NLP there are a few other activities on standardization and guidelines, for instance, the Automatic Content Extraction (ACE) guidelines, SpatialML, or TimeML [**Wahl et al., 2011**].

Automatic Content Extraction Guidelines. Automatic Content Extraction is a research topic started in 1999. It targets to develop technologies

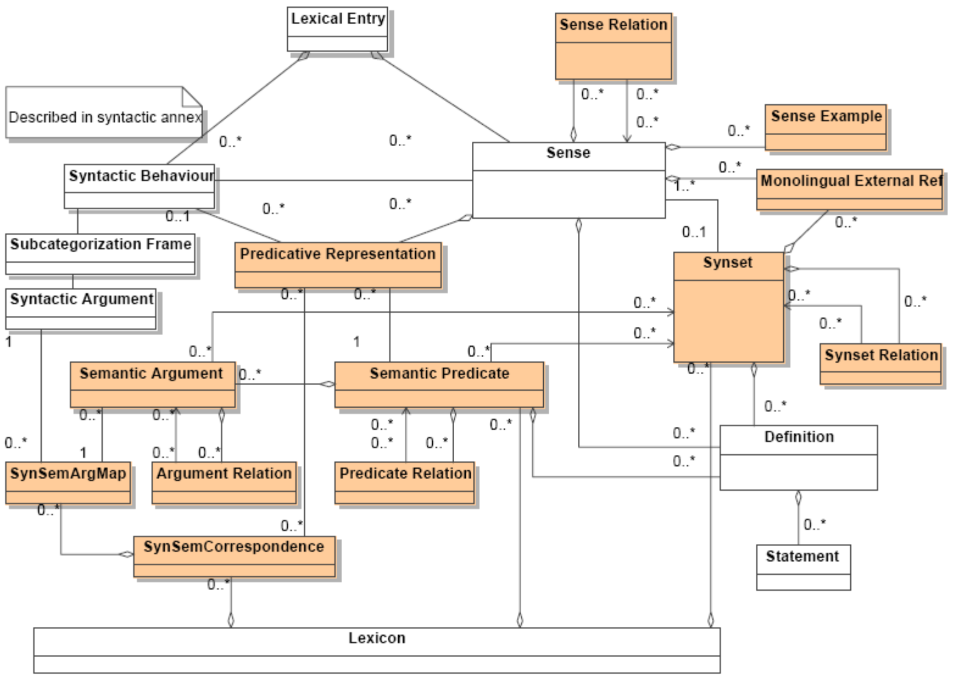


FIGURE 6. LMF Semantic Package [**ISO, 2008**]

$$\begin{bmatrix} FEATURE_1 & value_1 \\ FEATURE_2 & value_2 \\ FEATURE_3 & \begin{bmatrix} FEATURE_{31} & value_{31} \\ FEATURE_{32} & value_{32} \\ \ldots & \\ FEATURE_{3k} & value_{3k} \end{bmatrix} \\ \ldots & \\ FEATURE_n & value_n \end{bmatrix}$$

FIGURE 7. TFS in Matrix Notation

for the automatic characterization and extraction of meanings from (natural) language resources. ACE can detect and characterize entities, relations between such entities, and events [**Wahl et al., 2011**].

SpatialML. SpatialML is an annotation scheme for marking spatial expressions in natural languages. SpatialML is XML notated with additional possibilities, e.g. to annotate country names or geographic coordinates. As an example, the Listing 2 below gives the SpatialML notation of the sentence “Mexico is in North America”. There, all annotated SpatialML expressions are displayed with brackets, and below the corresponding XML markup is shown [**MITRE, 2010, Wahl et al., 2011**].

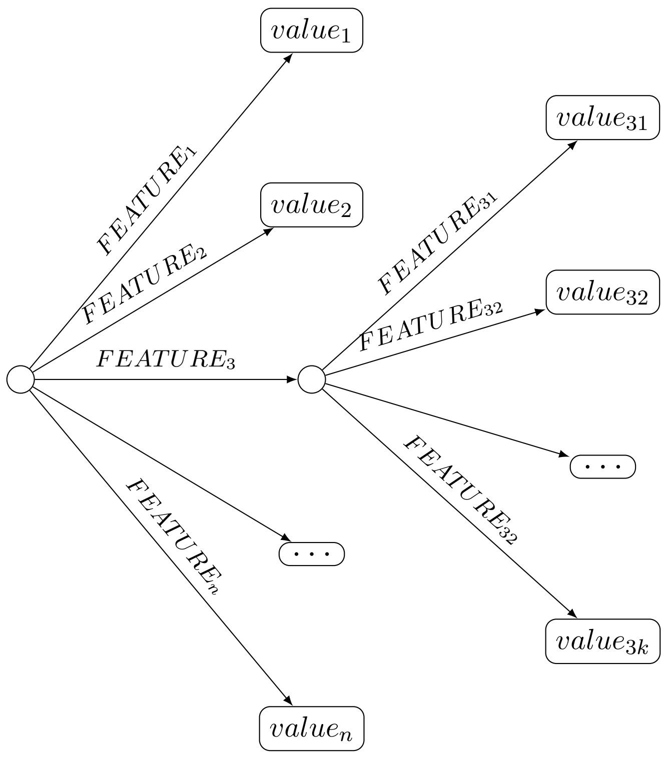


Figure 8. TFS in Graph Notation

$$\begin{bmatrix} ORTH & 'love' \\ SYNTAX & \begin{bmatrix} POS & verb \\ VALENCE & transitive \end{bmatrix} \end{bmatrix}$$

Figure 9. The Word "love" in Matrix Notation

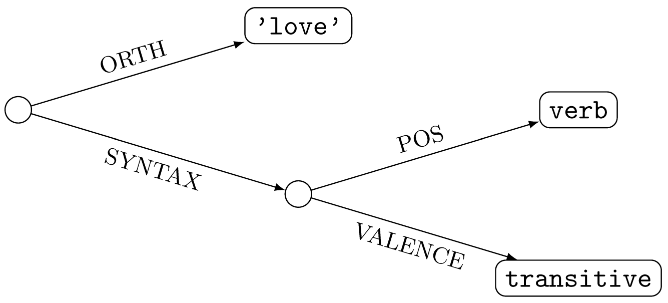


Figure 10. The Word "love" in Graph Notation

```
[Mexico] is in [North America]

<PLACE type="COUNTRY" country="MX" form="NAM">
  Mexico
</PLACE>
<PLACE type="CONTINENT" continent="NA" form="NAM">
  North America
</PLACE>
```

Listing 2. Example of SpatialML

TimeML. Events or temporal expressions in natural language resources can be formally specified by TimeML. TimeML can notate time stamps of events and even ordering of events (lexical properties of ordering or discourse). Additionally, it can handle difficulties of time reasoning within specific texts. It is not only made to mark temporal expressions. It brings together time expressions to event predicates [**Antoniadis et al., 2013**]. Notation of TimeML uses XML. The following Listing 3 gives the XML coded TimeML marking of the phrase: "John left between Monday and Wednesday" [**Sag et al., 2003, Wahl et al., 2011**].

```
John left between
<TIMEX3 tid="t1" type="DATE">
  Monday
</TIMEX3>
and
<TIMEX3 tid="t2" type="DATE">
  Wednesday
</TIMEX3>
```

LISTING 3. Example of TimeML

3.5. E-Learning Standards

Several interest groups deal with e-learning standards, for instance, ISO, European Committee for Standardization (CEN), Deutsches Institut für Normung (DIN), Institute of Electrical and Electronics Engineers (IEEE), or IMS Global Learning Consortium (IMS). The ISO/IEC JTC 1/SC 36[27] is a sub-committee of the Joint Technical Committee of ISO and International Electrotechnical Commission (IEC). It works on Information Technology for learning, education and training. Several work groups (WG) deal with specific topics within the area, for example, WG 1 defines a domain-specific terminology, WG 6 cares about integration of platforms and services, or WG 8 handles questions about interoperability and learning analytics. At the sub-committee's Web site the scope of work is defined as:

> "Standardization in the field of information technologies for learning, education, and training to support individuals, groups, or organizations, and to enable interoperability and re-usability of resources and tools."[28]

Besides technology, sub-committee 36 has also published a standard regarding quality issues of e-learning. Part 1 (General approach) of "Information technology – Learning, education and training – Quality management, assurance and metrics" [**ISO, 2005a**] is already finished. Part 2 (Quality model), part 3 (Reference methods and metrics), and part 4 (Best practice and implementation guide) are currently in progress.

[27] http://www.iso.org/iso/home/standards_development/list_of_iso_technical_committees/iso_technical_committee_participation.htm?commid=45392

[28] http://www.iso.org/iso/iso_technical_committee?commid=45392

CEN deals with both, technical and non-technical issues: the CEN Technical Committee 353 (CEN/TC 353)[29] aims to "Produce standards in the field of information and communication technologies relating to learning, education and training ..." and CEN/TC 428[30] deals with "Digital competences and ICT Professionalism".

The IEEE Educational Activities Board (EAB)[31] develops recommendations and policies in education specifically meant for IEEE engineering and scientific communities.

DIN groups mainly work on quality of learning environments and didactics. Typical topics are quality management, quality assurance, or process models.

IMS[32] developed a specification called IMS Content Packaging (IMS CP) with the goal to exchange learning data between learning systems. It is possible to import packages (and export, respectively), or to aggregate packages (and dis-aggregate, respectively).

According to [**Ehlers and Pawlowski, 2006**], we can distinguish two kinds of standards, firstly, learning technology standards, and secondly, e-learning related standards. Non-technical standards are, for instance, DIN PAS 1032 (eduction with focus on e-learning), DIN PAS 1037 (quality specifications for distance learning providers), DIN PAS 1068 (description of educational offers), DIN PAS 1069 (reference process model for quality management), ISO 9241 (ergonomics of human-computer interaction), or ISO 19796 which is a "framework to describe, compare, analyze, and implement quality management and quality assurance approaches" [**ISO, 2005a**].

Here, a brief overview of the most important technical standards is given. For more details, we refer to [**Ehlers and Pawlowski, 2006**] and [**Hoel et al., 2010**].

Learning Object Metadata. Learning Object Metadata (LOM)[33] describes learning content and learning resources. The basic metadata structure consists of nine main categories [**IEEE, 2002**].

(1) general: general information describing the learning object
(2) life-cycle: history and current state of the learning object
(3) meta-metadata: information about the metadata instance itself
(4) technical: technical requirements and technical characteristics
(5) educational: educational and pedagogic characteristics
(6) rights: intellectual property rights and conditions of use
(7) relation: relationship between the learning object and other related learning objects
(8) annotation: comments on the educational use of the learning object
(9) classification: describes this learning object in relation to a particular classification system.

[29]`https://standards.cen.eu/dyn/www/f?p=204:7:0::::FSP_ORG_ID,FSP_LANG_ID:580446,25&cs=1FE52F5B0EA5ABBBB805EAFDDC3F308A5`

[30]`https://standards.cen.eu/dyn/www/f?p=204:7:0::::FSP_ORG_ID:1218399&cs=1600F0DD849DA04F3E3B900863CB58F72`

[31]`http://www.ieee.org/education_careers/education/eab/index.html`

[32]`http://www.imsglobal.org/`

[33]`https://standards.ieee.org/findstds/standard/1484.12.1-2002.html`

Each of these main categories can additionally have sub-categories in a hierarchical way. In general, the metadata is added to the learning objects using XML notation. The following Listing 4 shows the corresponding namespace declaration [**IEEE, 2005**].

```
<lom xmlns="http://ltsc.ieee.org/xsd/LOM">
...
</lom>
```

Listing 4. LOM Namespace Declaration

For example, the main category "technical" consists of sub-elements format, size, location, requirement, installationRemarks, otherPlatformRequirements, and duration. As an example, in the Listing 5 the technical requirement of a "Flash Player plug-in" for "other" platforms is notated [**IEEE, 2005**].

```
<?xml version="1.0" encoding="UTF-8"?>
<lom xmlns="http://ltsc.ieee.org/xsd/LOM">
  <lom:general>
  ...
  </lom:general>
  ...
  <lom:technical>
    <lom:requirement>
    ...
    </lom:requirement>
    <lom:otherPlatformRequirements>
      <lom:string language="en">
        Plugin: Flash Player
      </lom:string>
    </lom:otherPlatformRequirements>
  </lom:technical>
  ...
</lom:lom>
```

Listing 5. LOM Example

Integrating Learning Outcomes and Competences. Integrating Learning Outcomes and Competences (INLOC)[34] aims at interoperable data exchange between ICT tools and services (e.g. university student information systems, recruitment systems, e-portfolio systems, or even social networks). The exchangeable data consist of information about learning outcomes and work competences. The existence of a globally unique identifier is a requirement for INLOC. On the one hand, this identifier makes it possible to distinguish different learning outcomes and, on the other hand, it allows to re-use a specific learning outcome. Each specific learning outcome is modeled by a definition (LOCdefinition) and a structure (LOCstructure). The LOCdefinition stores information like the identifier, title, description, level, credit, and the topic. The LOCstructure defines the structure of associated learning outcomes. For INLOC, there exist bindings for XML, RDF, and JSON.

[34]http://www.cetis.org.uk/inloc/Home

The following Listing 6 gives the representation of LOCdefinition in the XSD schema (taken from the INLOC Web site).

```
<xs:complexType name="LOCtypeLOCdefinition">
  <xs:complexContent>
    <xs:extension base="inloc:LOCtypeLOC">
      <xs:sequence>
      <xs:element name="primaryStructure"
        type="xs:anyURI" minOccurs="0" maxOccurs="1"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

Listing 6. INLOC Definition Example

Within the LOCdefinition a simple content description can look like shown in Listing 7, taken from the INLOC Web site.

```
<LOCdefinition id="http://example.eu/ids/A.2">
  <title>Service Level Management</title>
  <description>
    Defines, validates and makes applicable
    service level agreements (SLA) and underpinning
    contracts for services offered. Negotiates
    service performance levels taking into account
    the needs and capacity of customers and business.
  </description>
</LOCdefinition>
```

Listing 7. INLOC Content Example

Sharable Content Object Reference Model. The specification of Sharable Content Object Reference Model (SCORM)[35] provides a comprehensive framework for the realization of re-usable e-learning content. For this purpose, the approach offers specifications for content packaging, the communication with the Learning Management Systems (LMS) at content package runtime, and the definitions of the navigation and process flow of a learning unit. Content packaging defines how individual components of a learning content can be packed into a standardized file. Therefore, the so-called Content Aggregation Model is defined. Learning resources (called Assets, i.e. audio files, video files, or images) are put together to independent Sharable Content Objects (like a Web site within a course). Then, several Sharable Content Objects of a specific course are aggregated to a SCORM Content Package, which indicates the whole learning content of a specific course. Each level of aggregation can be tagged by meta-data using the above mentioned standard of LOM. In addition to the content packaging, the SCORM defines a Runtime Environment which is a standardized interface for the communication between learning contents and the learning management system. The Runtime Environment also allows to record learning activities of users. Runtime Environment specifications allow the

[35]`http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=53535`

execution of a content package within a compatible learning environment [**ISO, 2009b, Bohl et al., 2002**].

Tin Can API. Tin Can API (xAPI)[36] is a rather new e-learning software specification, currently available in version 1.0.2. Same as SCORM, xAPI makes it possible that learning content can exchange data with the Learning Management Systems. Additionally, it allows to gather different types of data of learners' experiences. Therefore, it is sometimes also called Experience API. The API captures activities data using a predefined vocabulary. Development of this vocabulary is done by the World Wide Web Consortium (W3C) group Experience API (xAPI) Vocabulary & Semantic Interoperability Community Group[37]. It is currently on a status of "awaiting public comment".

According to [**Hoel et al., 2010**], from above mentioned specifications and standards, only a few are commonly and frequently used. For instance, on content level, the IEEE LOM, and on level of management specifications, the SCORM and the similar IMS CP have high maturity.

3.6. Learning Software Architectures

The term "Software architecture" has been defined by Bass et al. as:

> "The software architecture of a program or computing system is the structure or structures of the system, which comprise software elements, the externally visible properties of those elements, and the relationships among them."
> [**Bass et al., 2012**]

The IEEE Learning Technology Standards Committee (LTSC)[38] works on systems interoperability in education. It deals with technical standards, and aims to publish recommendations for practical usage of e-learning. In the field of software architecture, it has developed a standard, i.e. the "Standard 1484.1-2003 – IEEE Standard for Learning Technology – Learning Technology Systems Architecture (LTSA)"[39], but this standard has been withdrawn and will not be continued any more.

Varying approaches of learning software architecture were developed. Learning software is seen as a specific application of software, therefore, different architectures are used, which are classical n-Tier architectures, Service-Oriented Architecture (SOA), Model Driven Architecture (MDA), or even proprietary solutions.

[**Bushehrian and Khaldar, 2011**] discuss in the work "An Extendable Software Architecture for Personalized E-Learning systems" their individual attempt of architecture that consists of three layers: Learner Interaction (LI), Labeled Transition Systems Explorer (LTSE), and a Recommender Component. Figure 11 shows their architecture for the Workflow Management System (WFMS). Workflow engines control activities of learners, while workflows are stored in an repository to analyze the learner

[36]`https://tincanapi.com/`

[37]`https://www.w3.org/community/xapivocabulary/`

[38]`https://ieee-sa.imeetcentral.com/ltsc/`

[39]`https://standards.ieee.org/findstds/standard/1484.1-2003.html`

actions. A recommendation service, based on a learner profile repository, allows personalization.

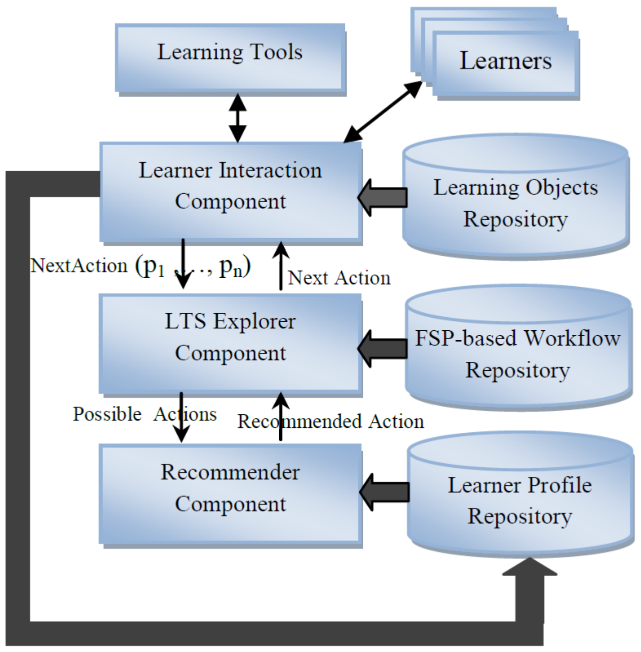


FIGURE 11. Architecture for the Workflow Management System (taken from [**Bushehrian and Khaldar, 2011**])

[**Antoniadis et al., 2005**] propose a prototype for their MIRTO project, which targets support for language teachers to get automatic exercises with the support of NLP. The authors describe their multi-layer architecture containing a function level, a script level, an activity level, and a scenario level. On function level, NLP functionalities happen, whereas the script level is a set of tools responsible for linking NLP functionalities with the didactic level. Here, teachers interact with the authoring system that allows to define learning scenarios with their sequences of activities.

[**Zhou et al., 2008**] describe in their work "The Design of Software Architecture for E-learning Platforms" an e-learning architecture, that mainly consists of four parts: the "Application Layer", the "Software Product Layer", the "Application Framework Layer", and the "Component Library Layer". The application layer can integrate already implemented e-learning software resources and software. The product layer includes several reusable manufacturing components to provide functionalities. The application framework is responsible for user interface management, user and access control management, content presentation, etc. Finally, the component library layer contains all software libraries (also from third parties) that implement e-learning purposes. Technically, the components interact using UDDI, WSDL, and SOAP protocols.

[**Zwickelbauer et al., 2015**] describe a technical architecture that was designed in a "Smart Learning" project. This architecture is more or less a classical 3-Tier architecture consisting of a user interface layer (learning apps, course editors, tests and exercises, etc.), a middleware layer (software for standardized data exchange, learning analytics software), and a data storage layer (learning objects repository, user database).

[**Chua and Lee, 2009**] introduce in their work "Collaborative learning using service-oriented architecture: A framework design" an SOA-based solution consisting of fours levels, which are the "Application UI", the "Business process", "Services", and "Service components". Services are meant to be re-used and, therefore, they have a generic and not a specific character. The application UI gives the user's perspective, and the business process level indicates the activities, which are implemented in the services and services components levels.

[**Dehbi et al., 2013**] describe in their work "A Model Driven Methodology Approach for e-Learning Platform Development" an approach based on MDA[40]. MDA was initially proposed by the Object Management Group[41] to allow applications to be portable. Thus, the same model can be realized on different platforms. Therefore, functional specifications are separated from implementation details. The basis of MDA are so-called computational independent models (CIM), platform independent models (PIM), and platform specific models (PSM). The CIM indicates the system requirements, the PIM is the abstract model, and the PSM is the concrete implementation on a specific platform. The authors introduce "LMSGENERATOR", which is a learning management system generator based on MDA paradigms.

[40]`http://www.omg.org/mda/`

[41]`http://www.omg.org/`

Part II

Towards the Intelligent Integrated Computer-Assisted Language Learning Environment

CHAPTER 4

Natural Language Processing Fundamentals

According to [**Jones, 1994**], the first phase of research in NLP started in the late 1940s, mainly with the focus on Machine Translation (MT). The first international scientific conference on MT took place in 1952, four years later the first Artificial Intelligence (AI) conference was held. In the late 1950s, NLP was correlated with Information Retrieval. The second phase (starting in the late 1960s) concentrated on meaning and knowledge representation. Work was "AI-flavoured and semantics-oriented" [**Jones, 1994**]. The third phase started a decade later, NLP focused on lexical approaches and grammar issues combined with logic-for-meaning representations. Jones mentioned the fourth phase (beginning in the 1990s): statistical language processing, data analysis like semantic classifications, and speech recognitions were new in research. Nowadays, NLP is present in several applications, for instance, in Intelligent Computer-Assisted Language Learning systems to support language learning like the proposed iiCALL environment.

4.1. Sub-disciplines of Linguistics

For implementing the iiCALL environment, relevant sub-disciplines of linguistics have to be taken into account, which are e.g. morphology, computational phonology, syntax and semantics, pragmatics, discourse, or dialogue [**Indurkhya and Damerau, 2010, Francopoulo et al., 2006**]. Computational Phonology works on representation and processing of phonological information by the use of Information Technologies. From a linguistic point of view, phonology deals with the characteristics of language sounds and their assembly in words and phrases. For example, phonology in a language learning environment can help to get better knowledge of pronunciations in languages.

In computational morphology processing of words and their written and spoken representations are relevant. Therefore, morphology analyzes the structure and the way how a word is constructed from its base form. For instance, using the English language, an "s" added to the end of a substantive might indicate its plural form. Characteristic morphological applications are spelling corrections or automatic hyphenation. In linguistics, syntax is the discipline which defines rules of how sentences are constructed. On the one hand, it guarantees the grammatical accuracy, and on the other hand it specifies the link between the meaning and the corresponding syntactic constructions. Here, examples of the English language are the rules: "A sentence can consist of a noun-verb-noun sequence" or "The noun before the verb indicates the actor" [**Wahl et al., 2011**].

In contrast, semantics is the science of identifying the meaning of words, phrases, or sentences. It addresses the way how to model the meaning of phrases or sentences, which might be derived from the meaning of their lexical items. Additionally, discourse analysis creates models for the meaning of discourses, which are sequences of sentences. From a different point of view, pragmatics tries to declare the meaning of dialogues (or linguistic communication) related to their context. Pragmatics and semantics are closely related with the variation of context inclusion, which might yield to additional differences in the meaning. For instance, the difference can be easily shown with the statement “It is cold in here” (taken from [**Antoniadis et al., 2013**]). The statement in its genuine meaning informs that at the place of the comment the temperature is low. In context, if it is known that at the place of utterance there is an opened window, the statement could implicitly have the intention that the receiver of the statement should close the window [**Wahl et al., 2011**].

4.2. Resources in Natural Language Processing

In NLP, different kinds of resources exist which are categorized by their linguistic richness. We differentiate between corpora, dictionaries, lexica, treebanks, propbanks, framenets, and wordnets [**Wahl et al., 2011**].

Corpora. Text corpora are machine readable samples of texts that typically occur in natural language. Usually, a corpus comes within a certain sampling frame. Examples are the radio news broadcasts of the British National Corpus or the spoken section of it. As specific requirements, corpora must be balanced and they have to be a representative textual collection. The data collection of a corpus is meant to investigate selective linguistic features. A corpus is called monolingual if it represents one single language. A comparable corpus is an accumulation of texts (in two or more languages) which are created in the same sample frame. A corpus is called parallel if it is a bi- or multilingual selection of texts where the translations are aligned at the sentence level. In contrast to text corpora, speech corpora are audio files or transcriptions with linguistic and phonetic annotations. A multimodal corpus is a combination of different communication modes like speech, hand gesture, facial expression, body posture, etc.

Typically, corpora are additionally tagged with several kinds of linguistic information which are obtained by analytical methods. The processed texts are parsed under different points of view. Shallow parsing indicates gathering morphological information like separation of text elements (e.g. words in sentences), description of syntactic word classes (nouns, verbs, adjectives, etc.), or named entity recognition. On the other side, deep parsing means analyzing text to gain further language information like semantic relations, semantics of discourses, knowledge representations, or constraints.

Parsing itself does not really add new information to a text, it emphasizes knowledge that is implicitly given. Different techniques and methods on texts result in annotations to be further processed.

Resources Creating Tools. The morphological analysis is used to recognize different parts of texts. It helps to distinguish various textual units like words or sentences. Unraveling texts into linguistic components (tokens) is called tokenization, such as punctuation, sentences, words, and numbers. Recognizing words in a text is referred as word segmentation. It is possible in some languages through finding spaces or word delimiters. It is not so simple for other languages such as Chinese and Japanese, which use delimiters for sentences but not for words. The discipline of separating different sentences in a text is described as sentence splitting. English, for instance, often splits sentences by punctuation (a full stop) but the character "." can also identify an abbreviation that can be used within a sentence (e.g. "etc." or "e.g."). The process of shortening a word to its base form is called lemmatizing. For instance, words like "goes", "gone", and "went" are all English words that relate to the lemma "go". Dictionaries generally use the lemma to represent corresponding forms of the word. The process of classifying words into lexical classes is called Part of Speech (POS) tagging, such as nouns, verbs, adjectives, etc. Named entities are objects that can be identified by a proper name like persons, cities, locations, etc. Determining such entities in texts is known as Named Entity Recognition (NER). Predefined lists are available for specific domains like lists of companies, countries or human names.

Dictionaries and Lexica. A Machine Readable Dictionary (MRD) is an electronically accessible dictionary. Moreover, a lexicon collects words and phrases with semantic information. To integrate an MRD and a lexicon in a software program and to optimize the access to them, they preferably occur in the form of a lexical database.

Treebanks, Propbanks. Corpora are created for different purposes. A Treebank is a specific text corpus that stores additional syntactic (and sometimes semantic) information. A Treebank differs from a POS-annotated corpus as it holds annotations of the syntactic structure (e.g. phrase structure or dependency relations) which can usually be represented as trees. A Propbank (or a proposition bank) is a corpus that is annotated with verbal propositions and corresponding arguments. It annotates the semantic roles of predicates within the text.

FrameNet. FrameNet is based on frame semantics, which is a theory of linguistics to analyze the meaning of texts. It is crucial for determining the meaning of a word to know the related situation in which the word is used. A semantic frame can be seen as a representation of such a situation. FrameNet is a lexical database storing examples of words and their semantic frames. The example shown in Figure 12 visualizes a part of the FrameNet using the tool FrameGrapher that is accessible via the FrameNet Web site[1].

Thesauri. Thesauri are collections of words that are interlinked according to their semantic similarity or relatedness. Thesauri can define classifications, synonyms, antonyms, etc.

[1]https://framenet.icsi.berkeley.edu/fndrupal/FrameGrapher

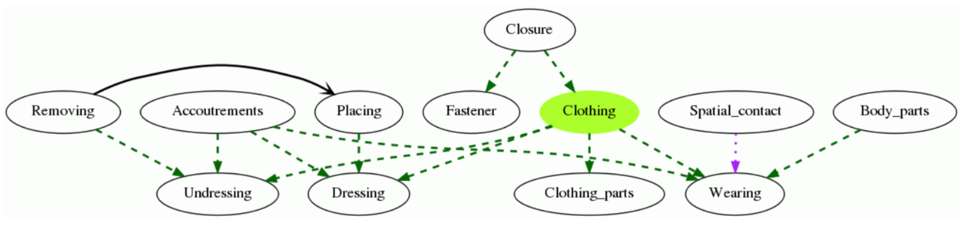


FIGURE 12. FrameNet Visualization Using FrameGrapher

Wordnets. The basic idea of a wordnet is the definition of synsets (i.e. set of synonyms) and their semantic relations. Besides the English WordNet[2], there are several activities to build wordnets in different languages like for German (GermaNet[3]), Japanese[4], or also for a collection of European languages (EuroWordNet[5]).

4.3. Toolkits for Natural Language Processing

If we search for a Natural Language Processing software, we will get numerous results such as "single parsers", "standalone annotators", and others. However, with respect to our research, NLP tools have to fulfill specific requirements. For example, it must be possible to integrate the tool or at least the functionalities of the tool. Moreover, the NLP software must be re-usable and extensible with additional functionalities. It is necessary to use a platform-independent coding language. As the learning of a language should be in a particular context, the tools must offer the option to access semantic data like ontologies. Numerous NLP toolkits are available. All those are assessed and compared regarding the contextual requirements of the language learning environment. The five selected options are Natural Language ToolKit (NLTK)[6], General Architecture for Text Engineering (GATE)[7], Ellogon[8], Clairlib[9], and Heart of Gold[10] [**Wahl et al., 2011**].

General Architecture for Text Engineering (GATE). GATE is basically a Language Engineering Open Source framework (an SDK), which has been implemented in Java using the Java's original component model (JavaBeans). Thus, it offers regular interfaces for re-usable modules. Resources of data are modeled in an object-oriented manner. Usually, it follows the TIPSTER architecture, which was invented by DARPA, the Defense Advanced Research Projects Agency [**Ralph, 1998**]. Whichever platform implements the Java Virtual Machine can install GATE. Additionally, GATE provides module integration in multiple languages such as Java, C++, Prolog, Lisp, Perl, or Tcl. GATE supports ontology interfaces on

[2]`http://wordnet.princeton.edu/`
[3]`http://www.sfs.uni-tuebingen.de/GermaNet/`
[4]`http://nlpwww.nict.go.jp/wn-ja/index.en.html`
[5]`http://www.illc.uva.nl/EuroWordNet/`
[6]NLTK Web page: http://www.nltk.org
[7]GATE Web page: http://gate.ac.uk
[8]Ellogon Web page: http://www.ellogon.org
[9]Clairlib Web page: http://www.clairlib.org
[10]Heart of Gold Web page: http://heartofgold.opendfki.de

the level of OWL-Lite. The GATE architecture is illustrated in Figure 13 [**Bontcheva et al., 2002, Wahl et al., 2011**].

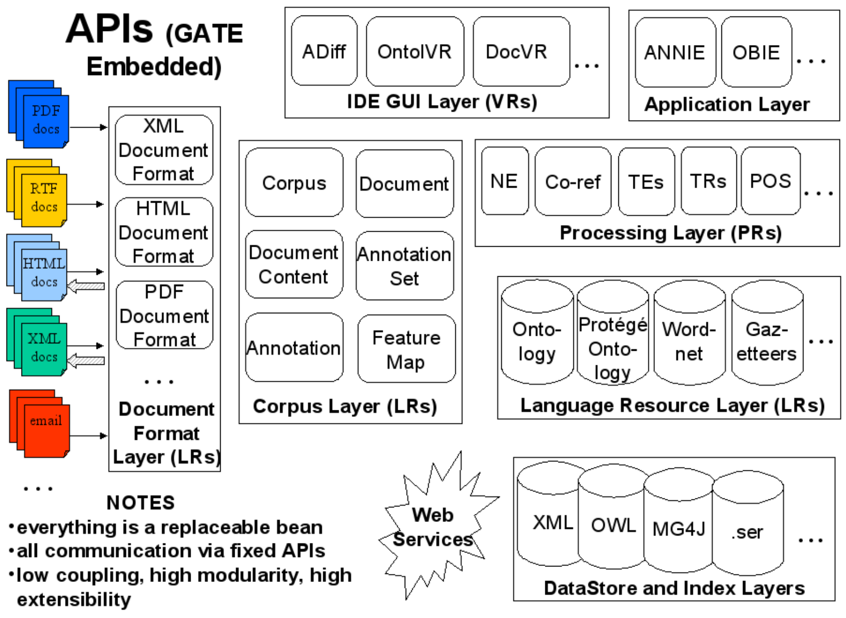


FIGURE 13. GATE Architecture [**Cunningham et al., 2013**]

For linguists (and not software developers), the GUI of GATE provides numerous operations for NLP and Language Engineering (LE). The GATE distribution also includes some plug-ins which include numerous LE operations such as sentence splitting, POS tagging, language identification, and more. On the other hand, CREOLE or "Collection of Reusable Objects for Language Engineering" for software developers comes with the GATE framework. The provided resources are JAR (Java-Archives) along with XML data configuration, and all these objects implement the plug-in rules for an easy re-use [**Cunningham et al., 2014, Wahl et al., 2011**].

Ellogon. Ellogon is a cross-platform and multi-lingual language engineering environment. It also provides a set of tools to manage, store and interchange textual data. It is written in the programming language C, and it offers extensibility for "Perl", "Python", "C", "C++", "Java" and "Tcl"-based systems. This software is available on "Microsoft$^{\text{TM}}$ Windows" and "Linux" systems. Its major functions are below: the core module of the software is in C and uses the extended TIPSTER data model version, same as GATE. It stores the linguistic information separately from textual data and keeps references to the original text as backup. It offers suitable infrastructure for textual data storage and related linguistic data. There is also an API for integrating it within different programs. Additionally, the GUI can be configured by the consumers in order to fulfill their specific requirements. The Ellogon architecture is shown in Figure 14 [**Wahl et al., 2011**].

Heart of Gold. Heart of Gold is a middleware software used to integrate NLP components. It is available on "Linux" platforms and has been

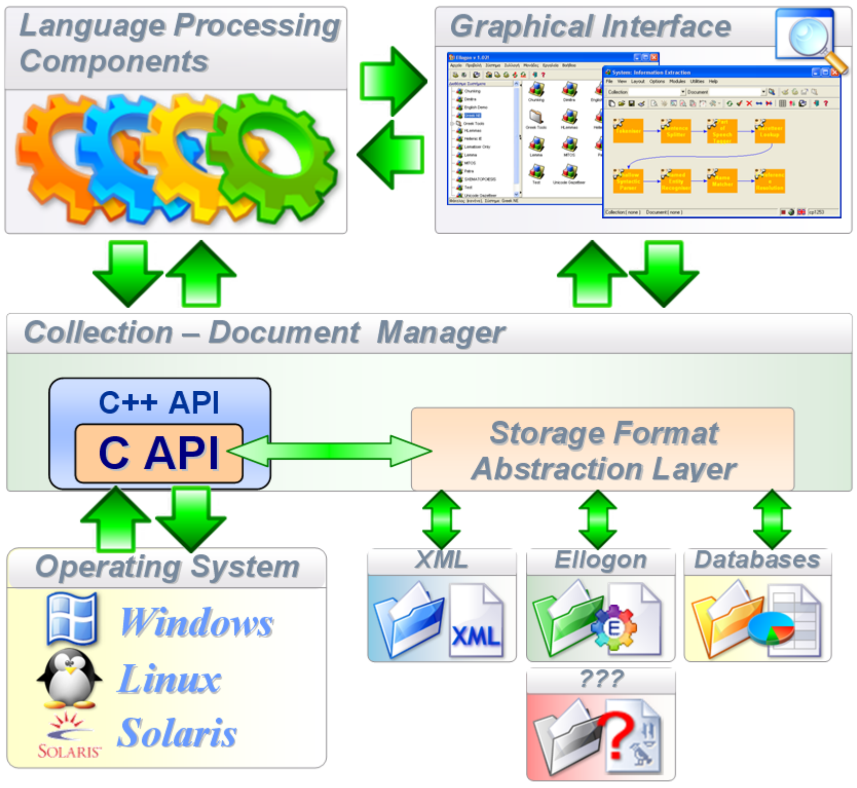

Language Processing Components
Graphical Interface
Collection – Document Manager
C++ API
C API
Storage Format Abstraction Layer
Operating System
Windows
Linux
Solaris
XML
Ellogon
Databases
???

FIGURE 14. Ellogon Architecture [**Petasis et al., 2003**]

implemented in Python and Java. The description language used in Heart of Gold is Robust Minimal Recursion Semantics (RMRS), which is coded with XML. However, the integration with non-RMRS natural language is also possible by annotation transformation. Moreover, it provides an API to implement individual features, and it allows platform independent communication via "XML-RPC". It is based on "XML", "XML-RPC", "XSLT", "XMLDB", and "XPath" technologies. The method summary of Heart of Gold is exhibited in Figure 15 [**Wahl et al., 2011**].

NLTK Natural Language ToolKit. NLTK is a group of Natural Language Processing libraries and programs, is implemented in "Python" and is available on several operating systems like "Mac OSX", "Microsoft^{TM} Windows", and "Linux". NLTK offers functions such as word frequency analysis, corpus processing, stemming, Part of Speech tagging, and collocation analysis. NLTK allows console processing but it does not provide a GUI [**Wahl et al., 2011**].

Clairlib. Clairlib is the abbreviation for Clair-library. Clair stands for one of the University of Michigan groups named "Computational Linguistics and Information Retrieval". It is basically an Open Source Perl modules collection for NLP, IR and Network Analysis (NA) operations. Clairlib is available on "UNIX", "Linux", "Microsoft^{TM} Windows", and "Solaris" system, too. It offers some ready to use modules, but the main purpose of Clairlib

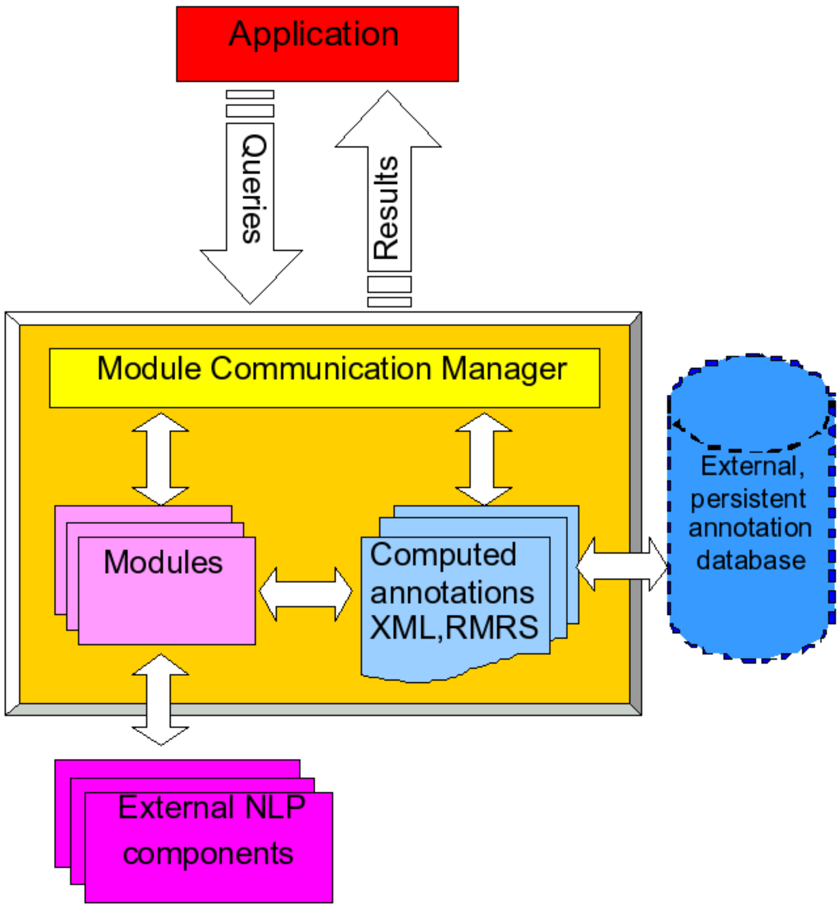

Application
Queries
Results
Module Communication Manager
Modules
Computed annotations XML,RMRS
External, persistent annotation database
External NLP components

FIGURE 15. Heart of Gold Middleware Architecture [**Callmeier et al., 2004**]

is to implement individual Perl modules working with Corpora. The major parts are “Clairlib-Core” and “Clairlib-Ext”. Clairlib-Core offers features like tokenization, summarization, document indexing and clustering, and Clairlib-Ext enables stemming, sentence segmentation, Web crawling and download, XML writing, parsing, and tree building [**Wahl et al., 2011**].

CHAPTER 5

Comparison of Natural Language Processing Toolkits

We evaluated the tools selected in Chapter 4 with respect to their potential to fulfill NLP tasks in an integrated language learning environment. The key variables for the comparison of Natural Language Processing toolkits are re-usability, extensibility, and functionality. The questions are, how many NLP functionalities do already exist that could be re-used for future processing inside the learning environment. Does software provide an API, SDK, or IDE, and what is the possibility of other application integration? For a comprehensive look at all the variables taken into account for comparison, we refer to Table 2 [**Wahl et al., 2011, Wahl et al., 2010**].

With respect to the requirements of the research for the language learning environment, the most suitable options are GATE and Ellogon. All the other options seem to have some excluding limitations. The benefits of GATE and Ellogon are: both fully support Unicode which is absolutely necessary to enable learning for all types of languages. Both of them also support a broad variety of data formats as input data like RTF, PDF, HTML, XML, and SGML. Additionally, their modules can be re-used and their new module development operations are well organized.

The most important benefit of the General Architecture for Text Engineering is the well designed software architecture which implements the "Java Component Model" applying JavaBeans. Thereby, it makes other architecture embedding easy. Hence, due to Java as main programming language, the platform is platform independent by default. We can also access GATE modules on smartphones if the Java Virtual Machine is running there, which at least Android smartphones do. GATE comes in different versions, one desktop version including a GUI to quickly try out some Natural Language Processing features. On the other hand, GATE Embedded is a version without having a GUI, to be integrated in a server environment allowing Web-based access by an application server. Even the functionalities of Semantic Web components especially for knowledge base integration is supported by GATE. In fact, GATE is the one and only evaluated software product that is by default capable of operating Web Ontology Language (OWL) on its level "OWL-Lite". Furthermore, for e-health applications, Unified Medical Language System (UMLS) mapping is also possible, remotely or locally [**Wahl et al., 2011, Wahl et al., 2010**].

TABLE 2. Summarization of NLP Tools

Criterion	**GATE**	**Ellogon**	**Heart of Gold**	**Clairlib**	**NLTK**
Executive summary	Framework (SDK) and graphical development environment for LE	Multi-lingual, cross-platform, general-purpose LE environment	Middleware for combining shallow and deep NLP components	Collection of Perl modules for NLP, IR, and NA	Collection of Python modules for NLP
Target group	Beginners to professionals in computer linguistics and system developers	System developers	Researchers	Researchers, Perl knowledge needed	Advanced in computer linguistics, Python knowledge needed
Extensibility, modularity	Modularly extensible, programmable	Modules for different tasks, less broad than GATE	Extensible by development of new components or applications	Modularly extensible, programmable	Extensible, programmable
Programming language	Completely in Java	C++ and Tcl	Java, Python, XML	Perl	Python
Unicode support	Unicode is default text encoding	Full Unicode support	Yes	Limited	Limited
Platform independent	Yes	No	Limited	Limited	Distributions for Windows, Mac OSX and Linux

Table 2. Summarization of NLP Tools

Criterion	**GATE**	**Ellogon**	**Heart of Gold**	**Clairlib**	**NLTK**
Support for mobile devices	Possible in a server environment and Java technology	Soon with single functionalities	No	No	No
Collaboration with other tools	Embedding is possible (by Java Beans), Tomcat integration	Bindings for C, C++, Java, Perl, Python, Tcl	Communication Interface via XML-RPC	No explicit interface	No explicit interface
Supported data formats	Plain text, HTML, SGML, XML, RTF, Email, PDF, MS Word	Plain text, HTML, SGML, XML, RTF, PDF	Plain text, HTML, XML	Plain text	Plain text, HTML, PDF
Ontology integration OWL	Yes	No	No	No	No
Parallel sessions	Yes	Yes	Yes	In principle, yes	No
Installation	Easy	Intermediate, requirements: Tcl/Tk and C++ compilers	Intermediate	Intermediate, Perl must be available on system	Intermediate, Python must be available on system
Documentation	Lots of documentations; user guide and developers guide	Lots of documentations; user guide and developers guide, component specifications	One single PDF document (about 80 pages)	Available online on a Wiki page	Lots of documentation, also available as book

TABLE 2. Summarization of NLP Tools

Criterion	**GATE**	**Ellogon**	**Heart of Gold**	**Clairlib**	**NLTK**
Database change possible	Yes	Yes	Yes	No	Yes
Automatic or manual semantic annotation	Both	Both	Both	Manually	Manually
GUI necessary for functionality	No	No	No	No	No
GUI	Yes	Yes	Yes	Not yet, promised for next release	No

CHAPTER 6

System Design and a Proper Development Process

An iiCALL system supporting Web-based and context-related language learning for arbitrary languages, being integrated in common working environments, and including NLP technologies is a highly complex e-learning system. Therefore, flexibility in adding functionalities, technical and semantic interoperability, extensibility and open development possibilities are prerequisites by default. Component-based Software Engineering offers theoretical aspects that may be applied to usefully contribute to the iiCALL development [**Heineman and Councill, 2001**].

6.1. System Design in the Field of E-Health

The Generic Component Model (GCM). [**Blobel, 2011**] explains the Generic Component Model which is one of the main theories behind the development of the Version 3 of the Health Level Seven International standard. He explains the complexity of e-systems (in a concrete example of e-health systems) by ideas taken from system engineering and system theory. To allow structural flexibility, functional flexibility, and multi-disciplinarity in system development, a kind of abstraction level is needed. Therefore, [**Blobel, 2011**] illustrates the Generic Component Model (GCM) as a cube where the axes are different views to the system (see Figure 16).

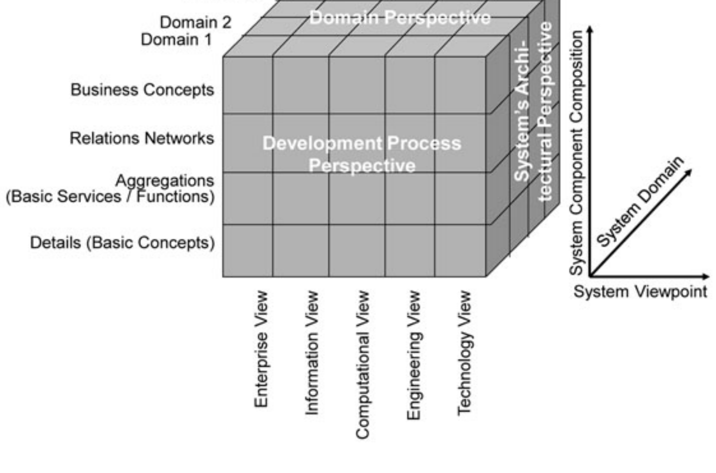


FIGURE 16. The Generic Component Model (GCM) [**Blobel, 2011**]

The x-axis is the System Viewpoint, which gets its different views based on the ISO 10764 (Information technology - Reference Model - Open Distributed Processing), i.e. Enterprise View, Information View, Computational View, Engineering View, and Technology View [**ISO, 2009a**]. The y-axis indicates the System Domain viewpoint. Here different domains, that may be represented within the systems, are assigned, for example, legal, administrative, technical, etc. views are relevant. Finally, the z-axis indicates the System Component Composition which shows the different levels of system granularity. Those levels start from Basic Concepts to Basic Services and Functions, Relations Networks, and Business Concepts.

Therefore, different perspectives on the cube display different system views. The x-y-area gives the Domain perspective, the x-z-area gives the Development Process perspective, and the y-z-area gives the System's Architectural perspective.

Ensuring interoperability is one of the main goals in system development, if flexibility, extensibility, and re-usability may be required. Therefore a common understanding of data and semantics of data is necessary. The need of a proper ontology (or proper ontologies) definition arises, where an ontology is seen as a representation of structure and behavior of a system. By the model of the GCM it is possible to identify needs for system interactions. Moreover, it allows to "zoom in" and to define partial semantic representations which are obligatory if one needs to take a look only on a sub-system. This may result in defining partial ontologies on domain-specific specializations. Blobel introduces the architectural approach to a system ontology by a layered approach using his GCM representation. Figure 17 (taken from [**Blobel, 2011**]) gives the example of an ICT ontology, which is actually a zoom into the ICT domain [**Blobel, 2011**].

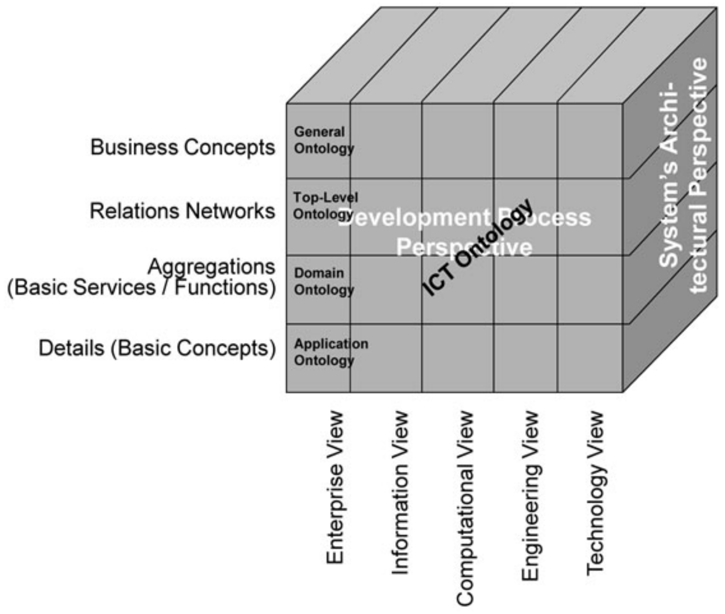


FIGURE 17. ICT Ontology in GCM [**Blobel, 2011**]

Here, for instance, the application ontology shall represent the basic concepts which are supported by the ICT solution on an enterprise level (with is the practical view for the business case). Interoperability between applications within a specific domain now can be realized by mapping the individual involved ontologies, which the domain ontology is responsible for. Top-Level ontologies moderate the integration of varying disciplines from varying domain ontologies.

Breaking down the GCM and ontology concept, mapping of ontologies on application level means interchanging data that is commonly understandable. A development process including all parties of involved actors is as important as an underlying data model which is generic and yields to a common semantic understanding.

6.2. Application Development in the Field of E-Health

Health Level Seven International (HL7)[1] is an international organization, which develops a set of standards for the exchange of information in healthcare. It is accredited by the American National Standards Institute (ANSI). From a practical point of view, the organization provides a collection of multiple message formats for data exchange between systems of different organizations at the application level for the aim of a semantic representation of clinical and administrative information. Therefore, the HL7 standard serves as a framework for the exchange of information in the healthcare system for clinical and administrative information. It is important to mention that the various healthcare facilities are very specific and individual. Due to the resulting complexity, i.e the diversity, processes, and models in healthcare, it is difficult to formalize one single uniform standard. Here, the fundamental concept of HL7 is examined. Accordingly, the methodology and design of messages for the efficient exchange of information are discussed [**Peixoto et al., 2010, Galler, 2015**].

The HL7 Healthcare Development Framework. HL7 is developing standards for a broad community of users in healthcare. For example, the Clinical Document Architecture (CDA) describes the coding, structure and semantics for an XML-based exchange of clinical documents. For instance, Structured Product Labeling (SPL) is a description of the structure and semantics of information that must be delivered together with medicines. So, the instructional leaflet becomes machine-readable [**Boone, 2011, Dolin et al., 2006**]..

However, here only parts of the "HL7 Version 3 Normative Edition" are relevant. The focus is on the methodology of the Healthcare Development Framework (HDF), and in particular on the underlying reference data model. Figure 18 (taken from [**Galler, 2015**] modified from [**Hinchley, 2007**]) provides a rough overview of the methodology in HL7 V3. It does not simply define a format for the exchange of messages but rather a model-based approach for the specification of such messages. Thus, HL7 uses suitable mechanisms to model a specific message type in a formalized manner for a particular application. As a result, a concrete message can be constructed

[1]`http://www.hl7.org/`

and implemented using this model. The UML is predominantly used for modeling [**Piho et al., 2015, Galler, 2015**].

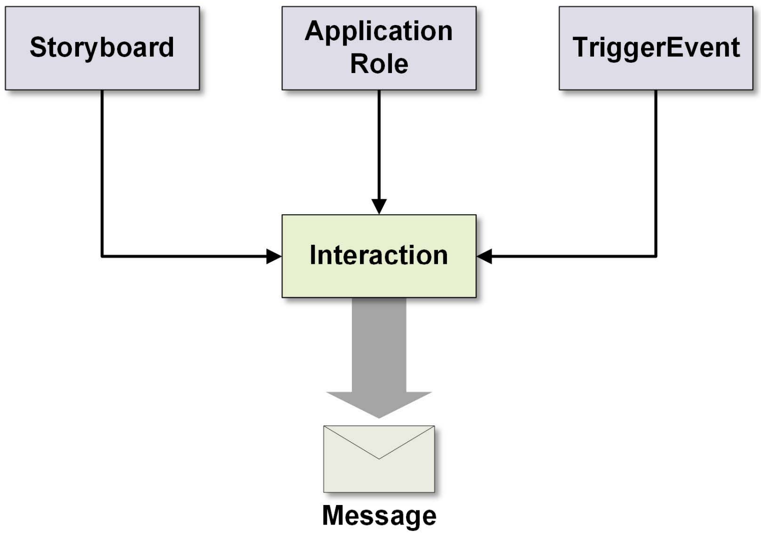


Figure 18. Overview of the HL7 V3 Message Description Method (after [**Hinchley, 2007**])

In Figure 18, storyboards essentially describe what happens to a user in a specific application from the user's point of view. This is more or less a use case in the notation of UML. Trigger events define events that cause a specific message to be sent. An interaction represents a unique link between a dedicated event and a message in the context of a particular application. Application roles describe the behavior of sender or recipient of a specific message. These fundamental elements are further discussed in detail below [**Galler, 2015, Hinchley, 2007**].

Storyboard. A storyboard is the beginning for the design of any message in HL7. It is a representation of events and situations in a chronological sequence to describe a concrete application in the healthcare system. Storyboards consist of a brief description and a specification that describe the interrelationship between involved parties. The standard also provides so-called storyboard narratives. Narratives are the descriptions of events that specify the context for interactions from the storyboard. Comparable to UML, it also specifies interactions between an actor and the system. Storyboards are characterized by a unique identifier, and in addition by a brief description. It contains a meaningful title as well as information on purpose and intent in text format. According to the UML interaction diagram, a list of the interactions is referenced there, and a detailed description of the interaction.

Trigger Event. A trigger event is a collection of one or more explicit conditions, which result in a message exchange between systems. These conditions must be present in a way to be recognized and processed by the system. In the standard each of these trigger events has to provide a name, a unique label, a description, and a hierarchically structured name. There are four types of trigger events:

- Interaction-based: the event is triggered by an interaction (e.g. the receipt of a message).
- State-transition-based: the event is triggered by changing a state.
- User-based: the event is triggered by a user.
- Unspecified: the trigger for the event is not assigned to any of the previous three types.

Application Role. An application role defines the behavior of a system, depending on the messages it receives or transmits. A system can also perform one or more roles. A role is used to define liabilities that must be fulfilled or implemented by the system. It defines the receipt of certain types of messages (receiver responsibilities). The formulation is carried out in an abstract manner. A system covers a certain application role when:

- It is able to send all message types provided by the application role in response to a defined event.
- It can receive all message types which are provided in the application role.
- It will meet the liabilities set for received message types.

Interaction. An interaction assigns the trigger elements to each other. A trigger event starts an interaction and prompts a message to be sent. The interaction references a storyboard, and correlates the user's view with the related requirements. By referring to one or more application roles, the liabilities for the recipient of the message are defined. Each interaction is uniquely determined by connecting a specific trigger event to a certain message type.

Message Design. Beginning with an interaction, a message type can be constructed in several steps, with which a particular message can be reproduced. For this purpose, HL7 V3 provides several concepts which are applied successively. The concepts are the following:

- Reference Information Model (RIM)
- Domain Message Information Model (D-MIM)
- Refined Message Information Model (R-MIM)
- Hierarchical Message Description (HMD)
- Message Type

Figure 19 represents an overview of the sequence of the individual steps in the design of such a message. These concepts are discussed in detail below.

Reference Information Model (RIM). The Reference Information Model indicates the generic data model from which all HL7 messages are derived. It consists of six core classes and further classes derived from them that represent all different domains in the healthcare system at a generic level.

The RIM is a collection of all the classes, relationships, and attributes needed to exchange information in healthcare. The UML class diagram is used for the description. The core of the model consists of six basic classes called the RIM Backbone, which are shown in Figure 20 (modified from [**Hinchley, 2007**]). The basic classes Entity, Role and Act represent

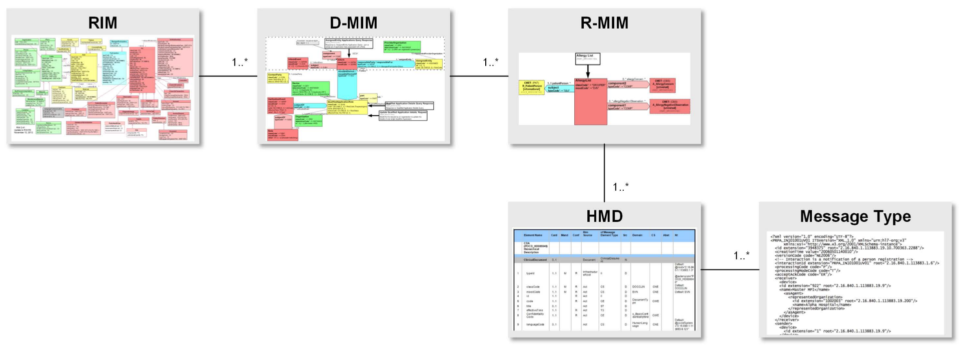


FIGURE 19. HL7 V3 Message Design [**Galler, 2015**]

objects, roles and actions from the healthcare system. These are linked to each other by the basic classes Participation, ActRelationship and RoleLink [**Schadow et al., 2006, Galler, 2015**].

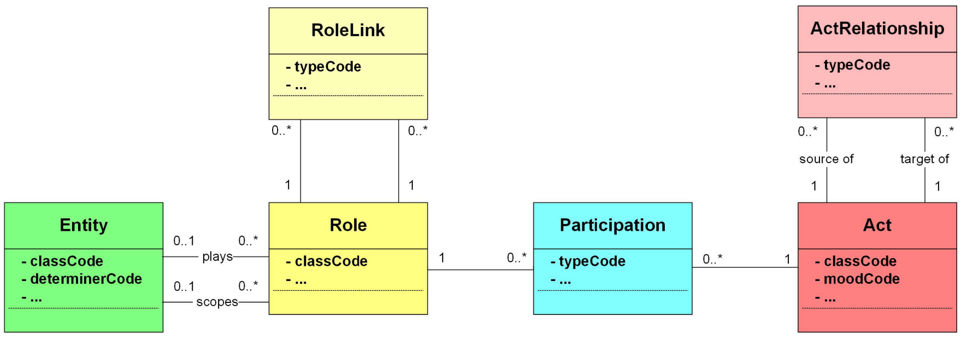


FIGURE 20. HL7 V3 RIM Backbone [**Galler, 2015**]

Starting from the three basic classes Entity, Role and Act, all other classes required for the information exchange are derived in an object-oriented way. So, the derived class inherits attributes and relations of the class. However, the derived class can define additional attributes. It is also important that the associations of the individual classes, together with their cardinality, are already defined at the top level in the RIM Backbone and are inherited from all derived classes.

For instance, a Person is finally derived from Entity. Therefore, it is a specialization of a Living Subject which is derived from Entity. A Person has some additional attributes, e.g the “educationLevelCode”, which gives education level of the person, or “addr”, which gives the person’s address (see Figure 21) [**Wahl and Winiwarter, 2013b**].

Thus, a role can indeed participate in several acts, whereas the Participation class itself can relate only one act and one role to each other. Special attention is also given to the attributes of the individual classes in the RIM Backbone, which are also called structural attributes. The attribute `classCode` is used by the classes Act, Role, and Entity and is a mechanism to explicitly specify the specialization in an attribute. The classes Participation, RoleLink and ActRelationship do not have a `classCode` because they are usually not specialized. The `moodCode` is used exclusively by the

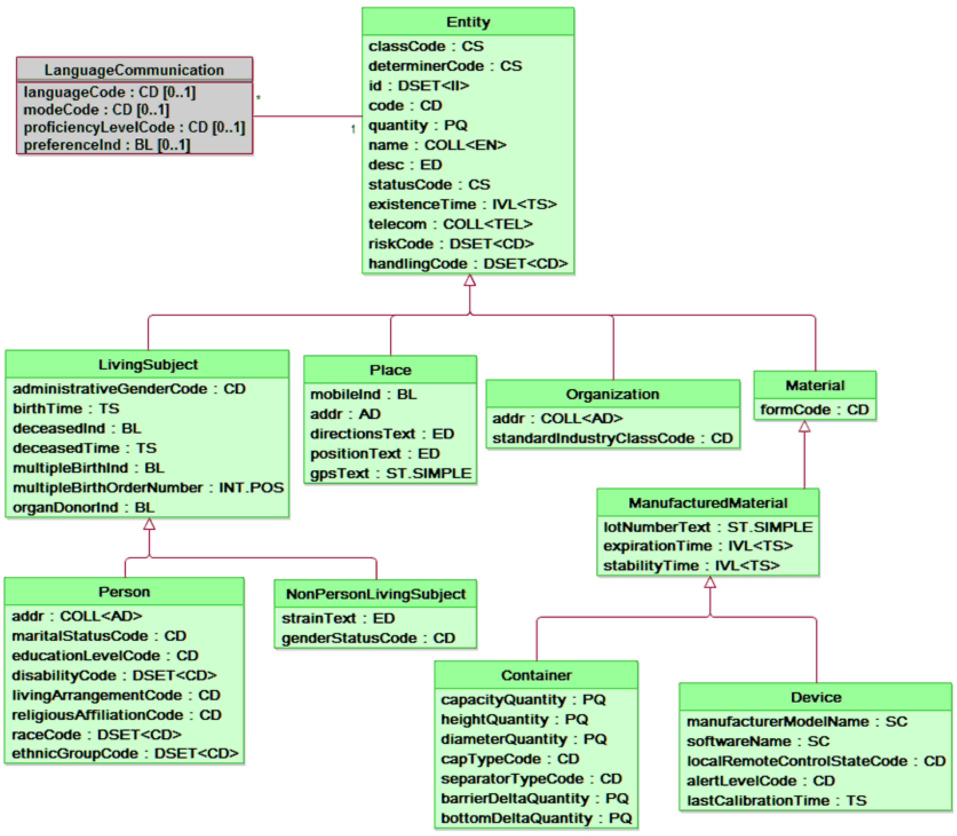


FIGURE 21. HL7 RIM Entity Classes [**HL7, 2016**]

Act class and opens up the possibility to record the nature of the process formulated with an act. For example, an action can be seen as a proposal (Proposal, `typeCode = PRP`) or as an intent (Intent, `typeCode = INT`). The combination of `moodCode` and ActRelationship allows a chaining of several acts. For example, the request for a laboratory result can be marked as an order (Request/Order, `typeCode = RQO`) and the presence of the result as an event (Event, `typeCode = EVN`). The relationship between these two acts is accordingly recorded through an ActRelationship [**Galler, 2015, Smith and Ceusters, 2006**].

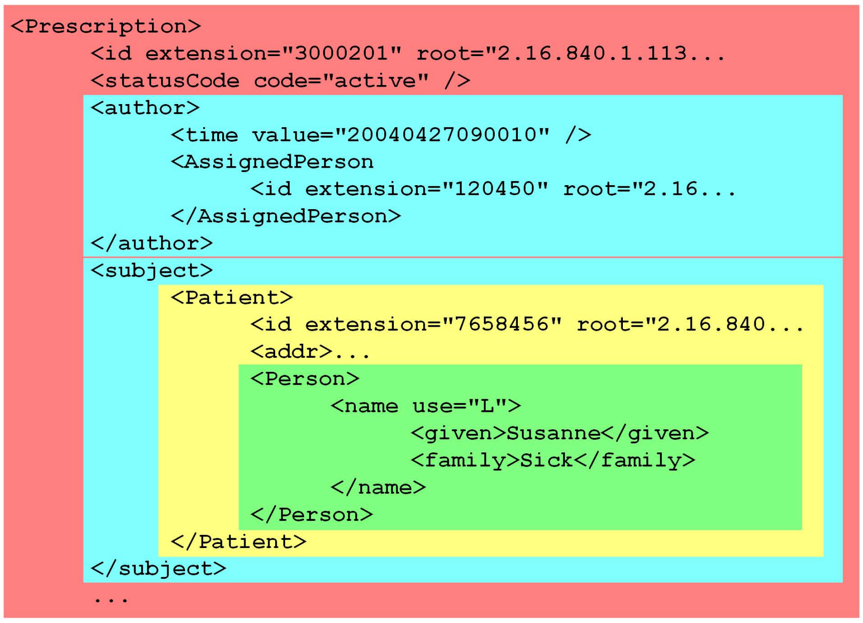
Domain Message Information Model (D-MIM). All relevant classes and attributes for a specific domain are selected from the RIM. So, the Domain Message Information Model is a reference model only needed for a specific domain, which shall cover all possible messages that can occur within the given domain.

Refined Message Information Model (R-MIM) / Constrained Information Model (CIM). The Domain Message Information Model is further refined to accommodate messages for a particular group within that domain. Base for the design of HL7 V3 messages is the RIM as explained above. From this, the classes and associations which are relevant for a specific field of application are selected or restricted. The result of this selection is a refined and restricted model, which is also called the Constrained Information Model. HL7 V3 performs this process in two phases. The result of the first phase is a so-called Domain Message Information Model, with which the RIM is adapted to a specific domain. In the second phase, the D-MIM is further restricted to an R-MIM. This is an even more specific model that finds application in a particular domain. A D-MIM covers the design of messages of a specific domain, whereas a R-MIM treats only one subset from a domain of that domain [**Luzi et al., 2010, Galler, 2015**].

Hierarchical Message Description (HMD). The Hierarchical Message Description (HMD) describes an alternative, flat representation of the R-MIM, from which further message types can be derived. The HMD represents an implementation-independent and structured representation of HL7 messages. For this purpose, the R-MIM is converted into a tabular structure. This can also be automated with appropriate tools. The result is a hierarchically structured list of classes from the R-MIM including their attributes. This is a one-dimensional representation of the two-dimensional model. An HMD can define several message types.

Mandatory attributes must always be assigned a value in a message. If this is not the case, the recipient cannot process the message. Particularly, structural attributes are therefore marked as mandatory. Required attributes are always present in the structure of a message. A transmitter, however, does not necessarily have to assign a value to it, but can also leave it undefined. In such a case, the recipient must have the competency to accept a default value for the affected attribute.

Message Type. Message types result from a further restriction of the HMD. From the message types defined in the HMD, concrete messages can now be derived. For this, HL7 V3 provides various Implementation Technology Specifications (ITS). These implementation rules define how to implement a message type for a specific technology. Implementation can also take place automatically by the use of appropriate tools. In particular, the representation of messages in XML has been established. For this purpose, the individual message elements from the HMD are mapped line by line to XML elements in an XML schema. Exceptions are structural attributes such as `classCode` and `moodCode`. These are represented in the schema not in the form of elements but as XML attributes. For predefined data types, the default provides a schema that can be referenced. The generated messages in XML format always use the same namespace. To check the validity of such XML messages, they can be validated against created XML schemata. Figure 22 shows an extract of the XML representation. For better traceability, consistent coloring is used for the individual XML elements. The entry-point class Prescription also represents the root element in the XML document. It contains the two subordinate elements `<author>` and `<subject>`, which are Participations. The `<subject>` again contains the Role of the patient as a subordinate element. A person named "Susanne Sick" takes this role in this message. In this case, the structure of the name (`<given>`, `<family>`) is defined by a data type from the standard [**Hinchley, 2007, Galler, 2015**].



```
<Prescription>
      <id extension="3000201" root="2.16.840.1.113...
      <statusCode code="active" />
      <author>
            <time value="20040427090010" />
            <AssignedPerson
                  <id extension="120450" root="2.16...
            </AssignedPerson>
      </author>
      <subject>
            <Patient>
                  <id extension="7658456" root="2.16.840...
                  <addr>...
                  <Person>
                        <name use="L">
                              <given>Susanne</given>
                              <family>Sick</family>
                        </name>
                  </Person>
            </Patient>
      </subject>
      ...
```

FIGURE 22. HL7 V3 XML Message (taken from [**Galler, 2015**] modified from [**Hinchley, 2007**])

Summary. Figure 23 shows the HDF by combining methodology, modeling, and message design.

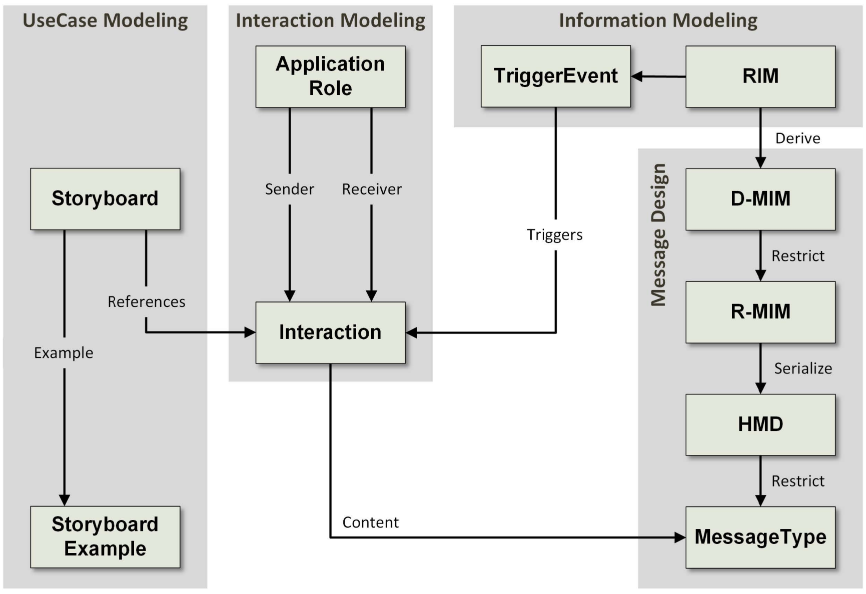


FIGURE 23. HL7 V3 Overview Including Methodology, Modeling, and Message Design [**Galler, 2015**]

The beginning is a storyboard, similar to the concept of UML use cases. Requirements and processes can be defined in a simple way in textual form.

Application Roles define the behavior and, in particular, the liabilities associated with the sending and receiving of messages. Trigger events document the cause of sending a message. These three elements are ultimately joined by an interaction. The RIM represents the starting point for the design of messages. D-MIM selects those classes and associations that are relevant to a given domain. By further restrictions, this model can be refined to a range-specific R-MIM. The HMD presents a one-dimensional description of this model in the form of a hierarchically structured list. Thus, the message is described in a manner independent of the implementation. From the resulting message types, the XML schemata are derived using the corresponding ITS, with the aid of which messages can be represented in the XML format. The great advantage of the top-down approach of this methodology is interoperability between different domains in healthcare, as well as between different sections within these domains. Further, this approach allows a design of messages without already defining the concrete implementation technology for the message exchange [**HL7, 2010, Galler, 2015**].

6.3. Comparison to the Field of Language Learning

When one takes a closer look at the language learning domain and the language learning e-learning system requirements of iiCALL, a variety of similarities can be identified. Firstly, from a very rough architectural point of view, it is an e-system that uses Web technologies to provide its services. Technical requirements of iiCALL are comparable, which are the inclusion of different sub-systems, the support of open and collaborative development, a high complexity due to the huge amount of languages, user skills, user preferences, and imaginable learning scenarios.

Same as in the e-health area, to allow structural and functional flexibility and to support multi-disciplinarity, technical and semantic interoperability is needed. Functionalities shall be implemented as services which allow reusability. Therefore, an abstraction of learning seems to be necessary.

Even the model as shown in Figure 16 appears to be applicable for iiCALL. Thus, the system viewpoint according to the ISO 10764 (Information technology - Reference Model - Open Distributed Processing) which describes an approach to object modeling to formally specify distributed e-systems in heterogeneous environments, generally defines the standard views (enterprise, information, computational, engineering, and technology) on systems and their environments. Within language learning systems, several different domains are relevant, for example, languages and learning theories, and naturally, legal, administrative, and technical aspects are appropriate in this regard. So, the system domain viewpoint must be taken into account in several characteristics. Finally, the system component composition, which indicates views to the different granularity of systems, is applicable.

6.4. Towards an iiCALL Development Process

Language learning can happen in many different methodologies, different techniques, and different learning processes. Learners may need specific multimedia support, hints from experts or coaches, automatic tests, or communication within a team to fulfill their learning tasks. Learning highly depends on the skill level of learners, so methods may need to change depending on a reached level of a learner. Thinking of implementing new features, different data structures for storage and communication may occur. A common understanding of data and semantics of data, flexibility, extensibility, and re-usability might be necessary properties to achieve a software for further development [**Wahl and Winiwarter, 2013b**].

In a similar way, comparable problems have been seen in the field of healthcare. In the beginning, different software vendors implemented very specific solutions which could hardly be adapted to different applications. Nowadays, technical and semantic interoperability has become more and more important. Thus, standardization like the HL7 Healthcare Development Framework and the underlying data models try to overcome those difficulties [**Blobel et al., 2006**].

Towards a Proper Generic Data Model. The HL7 aims to define standards to handle all possible elements needed for developing software in the healthcare sector. In the Version 3 of the HL7, the generic approach using the RIM was introduced (see page 47). The core data model consists only of six core classes, it is called the RIM Backbone (see Figure 20 on page 48) [**Wahl and Winiwarter, 2013b, Zhang et al., 2016**].

- Entity
- Role
- Participation
- Role Link
- Act
- Act Relationship

In [**Wahl and Winiwarter, 2013b**], we adapted the idea of the HL7 Reference Information Model and introduced the iiCALL Generic Data Model (GDM). The initial version showing the core classes of the iiCALL GDM is presented in Figure 24.

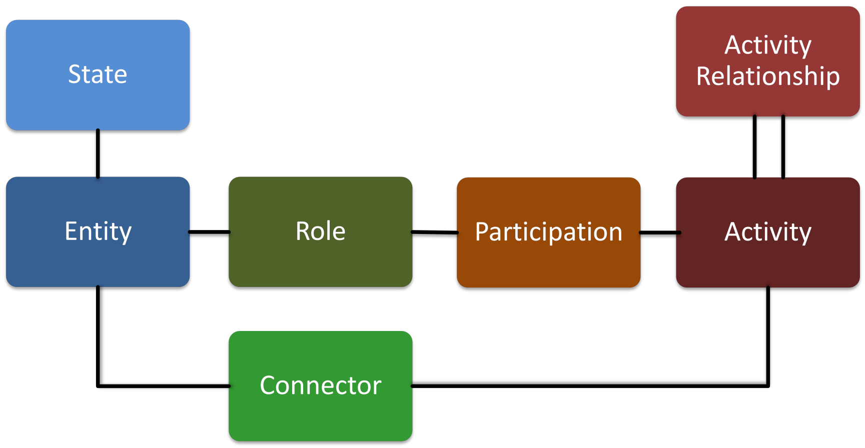


FIGURE 24. Core Classes of iiCALL GDM Draft Version

The Generic Data Model had some intersections with the HL7 RIM Backbone and two additional classes:

- Entity
- Role
- Participation
- Activity (comparable to Act in HL7 Reference Information Model)
- Activity Relationship (comparable to ActRelationship in HL7 Reference Information Model)
- State
- Connector

Entity is again a class for holding all kinds of objects that may occur in language learning. Entities participate in different activities, having a specific role. By both, Activity and Activity Relationship, process steps can be modeled, which makes the need for a specific workflow engine obsolete. The class State holds any kind of state an entity can have, e.g. a language skill level. The Connector class is responsible for connections to any standards used in the learning process, e.g. TFS, Lexical Markup Framework, SpatialML, or TimeML [**Wahl et al., 2010, Wahl et al., 2011**].

Towards a Proper Development Process. In healthcare software development, the HL7 HDF consists of modeling suggestions and process definitions for application development. An adaption of the framework we originally introduced in [**Wahl and Winiwarter, 2013b**], which is called the generic data model for iiCALL. For the whole process from definition of storyboards till the construction of SOA XML messages, we refer to Figure 23.

Adapting the HL7 HDF to the needs of language learning, we describe the first idea of the iiCALL development process. It contains four steps, which are the Concept phase, the Mapping phase (mapping to the iiCALL GDM), Adaption phase, and the Implementation phase (see Figure 25) [**Wahl and Winiwarter, 2013b**].

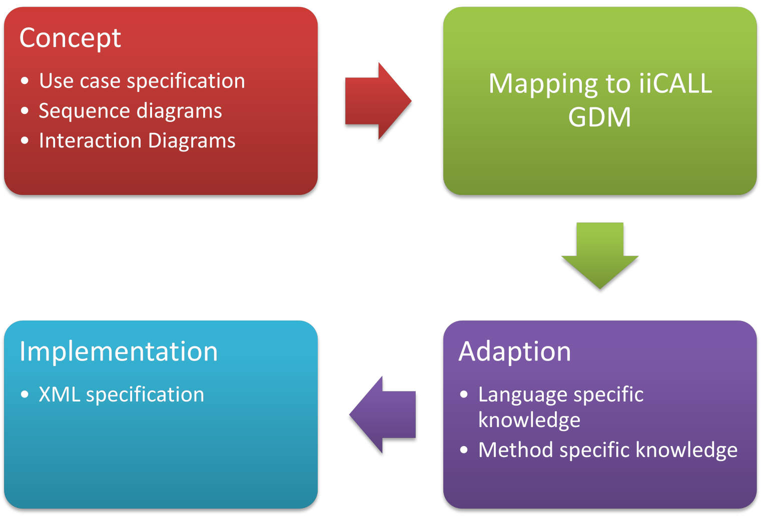


FIGURE 25. The iiCALL Development Process Draft Version

As in HL7, UML is used in the conceptual phase by use case modeling and verbal and formal specification. When specification is done, mapping to the Generic Data Model happens. It might happen while prototyping that subclasses of the GDM-Core have to be added due to new functionalities. Domain-specific adaptions follow, which means to integrate method-specific or language-specific knowledge. Finally, XML messages are generated to support the Service-Oriented Architecture.

Part III

Implementation

CHAPTER 7

A First iiCALL Architecture

It is important to note that the iiCALL system supports different levels of learners. For example, the level of learning begins from the beginner level and ends in the form of advanced learning. This yields to different understanding and dynamics of the integrated learning and different understanding how, for instance, a Web browser plug-in utilizes the browser's content as data for the respective purposes. Functional requirements for the iiCALL environment are achieved by use case driven requirement engineering.

7.1. Exemplary Use Cases

To be able to customize the iiCALL system for personal needs, the user must get the possibility to store individual information and preferences. Such a learner profile includes the mother tongue, additional languages the user is able to speak or understand, prior knowledge, and expertise in learning languages. Areas of interest and language experiences from a professional background may ease the assessment of learner level within a specific context. The user profile is stored on the client side, or it can be stored centralized on an iiCALL server. Entry level tests help the user to evaluate himself. For this purpose, the user can try a collection of predefined cloze tests. These tests are presented as multiple choice tests where one out of four possible answers has to be selected. Questions include different language topics like grammar, the right use of prepositions, word orders, or tenses. By the percentage of correct answers the user is classified in different levels. Up to 60 percent, one is called a "Beginner". In the range from 60 to 90 percent, one is classified as an "Intermediate", and all users who score more than 90 percent are categorized as "Advanced" [**Wahl and Winiwarter, 2011b**].

Beginners. The strategy to deal with the various types of students is different. In the case of the initial level, the vocabulary is very important. The introduction of the vocabulary trainer helps the learners to grab knowledge about the words. Vocabulary training is more or less correlated to the theory of Behaviorism (see Table 1 on page 17). Vocabulary training in the Web browser is done in a way that a text of the opened Web site is taken. The first step in this regard is to ascertain the language of the text. For this purpose the use of NLP is crucial. Additionally, the domain of the text shall be recognized. Therefore, available semantic models are checked for fitting the actual text. If language and domain match the learner's preferences, the system randomly chooses ten words taken from the source text, uses lemmatizing to determine the base form, and returns one correct and three incorrect translations as a single choice questionnaire. The learner tries to answer the questionnaire, after finishing, he gets presented an analysis showing correct answers compared with his own answers. For later re-do and look-up, all logging of tests is stored locally.

Intermediate. In case of intermediate level, there is the process of the "Cloze test" that builds on the idea of Cognitivism. The students are given the chance to select the text which seems appropriate to them. The system presents a fitting cloze text to the learner. The learner can ask for hints which are, for example, grammar rules, correct usage of tenses, or word forms of irregular verbs. There is the option of team work in this regard. In team work, native speakers of a particular language are also available for help.

Advanced. Following the idea of the theory of Constructivism, use case "Social translation" is suitable for advanced learners. Learners form virtual teams with the goal to translate a specific text from a source language to a destination language. The text itself can again come from a currently viewed Web site or can be suggested by the user. Learners learn implicitly by virtual discussions and guidance of native speakers.

Specific learner groups. The use case "Simple phrases for communication" might be useful for travelers or tourists. This group of learners need communication skills and phrases (and words) applicable in specific situations. Also a code of behavior might be very important. For example, to nod one's head can mean both, agreement or disagreement, depending on the culture or location. Greeting, asking for directions, or thanking are here as important as, for instance, ordering food in restaurants or hiring accommodations. To virtually simulate situations, multimedia support is essential.

The use case "Specific domain terminology" is intended to improve language skills within specific domains. Learners are supposed to already have a good knowledge of a specific language, but their goal is to get a deeper insight in the specific terminology of a domain. Engineers, chemists, nursing staff, mechanics, etc. might be applicable user groups.

7.2. Functional Requirements

Functional requirements of the iiCALL platform result from the main definitions of iiCALL and from several learning issues. Firstly, the iiCALL platform must be "integrated", which means that learning happens within usual working environments, at least Web browsers or email clients. Output devices can vary, so also tablets, smartphones, etc. shall be able to be used. Standardized communication and interaction by standard interfaces are needed. Language learning shall not be limited to a few languages, arbitrary language support implies complete support of the Unicode standard. Secondly, iiCALL shall be "intelligent" in the way that NLP, AI, and semantic technologies are used.

It has been observed that certain people feel more comfortable if learning is supported by visual learning materials like pictures and videos, while others prefer acoustic or textual materials. Multimedia supports the better understanding of languages and improves the learning progress.

Collaboration might improve learning results. Social and collaborative learning are established methods. In order to cater this need, the iiCALL system has the capability to interact with social media networks. Given the variety of the learning strategies and methodologies, an integrated approach shall be implemented.

The iiCALL platform follows the core principles of being an intelligent, integrated and Web-based environment that supports language independent and context-related language learning facilities with respect to different learning types. Language learning happens within a variety of learning scenarios, which can be supported, for instance, by Intelligent Language Tutoring Systems (ILTS) or Automatic Test Generation (ATG). For individual needs fitting different learning types the system has be be configurable, on the one hand with respect to learning flow of events and, on the other hand, with respect to individual preferences of learners. Storing learner (user) data via user profiles might be helpful, be it on the user interface (client) side as well as on the centralized server side [**Wahl and Winiwarter, 2011a**].

7.3. The First System Overview

Based on the comparison of different NLP toolkits (see Chapter 5 on page 39), GATE was selected to be the most suitable candidate for providing NLP functionalities for the iiCALL environment. GATE offers a variety of re-usable NLP tasks, individual functionalities can be easily implemented through the plug-in concept of GATE. In addition, processing of semantic data and language resources are advantages. Technically, GATE can be extended, and it allows to be integrated in a server environment, which is essential for Web applications. GATE is implemented in Java, so, Java has become the technical choice of programming language of the iiCALL system. With respect to further and collaborative developments, we will license iiCALL under a proper Open Source license [**Wahl and Winiwarter, 2011a**].

Figure 26 gives the first draft overview of the initial iiCALL software architecture. A server-side software framework builds the core part of the iiCALL environment (in Figure 26 noted by "1"). Here, an application server (i.e. an Apache Tomcat®[1]) handles Web communications. Java Servlets with the support of the Java Spring Web framework[2] are conceived. GATE Embedded runs within the Tomcat® environment, offers NLP tasks and provides access to several language resources (noted by "2"). Individual NLP tasks can be implemented following the Java component model (JavaBeans). Interfaces to semantic data are possible via the iiCALL Software Framework as well as via the GATE framework. Currently, PostgreSQL[3] is favored. Workflow engines for definitions of learning processes for learning scenarios are noted by "4". Clients are supposed to communicate with the iiCALL system via Web access (noted by "3"). Such clients

[1]`http://tomcat.apache.org/`
[2]`https://github.com/spring-projects/spring-framework`
[3]`https://www.postgresql.org/`

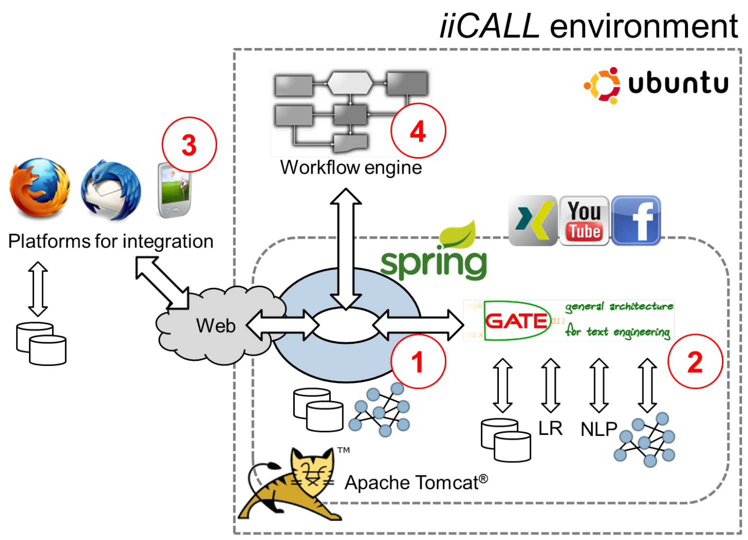

iiCALL environment
ubuntu
3
4
Workflow engine
Platforms for integration
spring
Web
GATE general architecture for text engineering
1
2
LR
NLP
Apache Tomcat®

FIGURE 26. First Draft Overview of the iiCALL Environment

might be Web browsers, email clients, etc. There, tiny databases (for example SQLite in the Firefox browser) have to be available to store user data [**Wahl and Winiwarter, 2011a**].

7.4. The First Firefox Plug-in: A Synonyms Look-up Tool

Integrability at the client side is firstly demonstrated by a tiny gimmick, which allows to look up synonyms. This tool is coded as an XML User Interface Language (XUL) plug-in in the Firefox Web browser which uses Asynchronous JavaScript and XML (AJAX) for communication. The tool listens to selected text of an opened Web site, sends the selection via XMLHttpRequest to the OpenThesaurus[4] service and then processes the obtained response. Normally, the response is a list of synonyms correlated to the original word. These synonyms are presented to the user within a fitting pop-up window. Because the OpenThesaurus supports only the German language, the plug-in can only deal with German inputs. Figure 27 shows the interface and the visualization of the tool which provides synonyms for the German word "Werdegang" [**Wahl and Winiwarter, 2011a**].

7.5. The Initial Prototype: A Context-related Vocabulary Trainer

The idea of the context-related vocabulary trainer is to offer vocabulary training within specific domains. It is important to note here that the learning type and methods are specific to the individuals. Hence, the psychological theory of learning has to play a crucial part in this regard. The application of the vocabulary training is built on the paradigm of the Behaviorism theory of learning. Users can define their interests and the languages they want to learn. In case the system identifies Web content fitting user

[4]`http://www.openthesaurus.de`

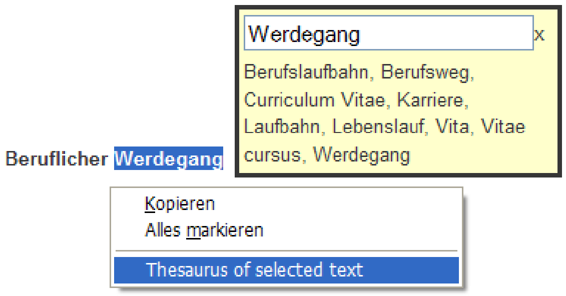

Werdegang
x
Berufslaufbahn, Berufsweg, Curriculum Vitae, Karriere, Laufbahn, Lebenslauf, Vita, Vitae cursus, Werdegang
Beruflicher Werdegang
Kopieren
Alles markieren
Thesaurus of selected text

FIGURE 27. A Helpful XUL Plug-in for Finding Synonyms for German Words Selected in the Web Browser

preferences, a proper vocabulary test is presented using words of the current web content.

Requirements and System Architecture of the Context-related Vocabulary Trainer. The main focus lies on context-related learning. A specific context is needed to ensure correct language usage. In health or in medicine, for example, nursing staff that has some language skills but needs to improve knowledge of medical terminology, can use the vocabulary trainer. Tourists need some improvements in languages specifically in the context of travel and daily communication. For business people finance and economics can be a possible context, engineers might need to train specific technical terms.

The vocabulary trainer is as a specific iiCALL system a Web-based e-learning solution. Thus, communication between client and server is done by sending requests from client side and awaiting responses from server part. Local database support is needed for user profiles or logging tasks [**Wahl and Winiwarter, 2011b, Wahl and Winiwarter, 2012b**].

Architecture and Data Flows. Derived from above conditions, the system overview with respect to data flows of the context-related vocabulary trainer is shown in Figure 28. The prototype consists of three components, which are:

- A plug-in for the Firefox Web browser
- A server component
- A GATE plug-in

Plug-in for Firefox. At client side, the Firefox plug-in allows the user to specify his personal preferences, which are at least language to be learned and context of interest. These user-specific data is stored locally. Future implementations allow synchronization with the server side to allow access to user profile data from anywhere. Additionally, here the vocabulary test finds a user interface. The client is implemented in XUL. It communicates with the iiCALL server via AJAX. Therefore, it sends XMLHttpRequests and awaits response from the system to process.

The Server Component. The server component implements functionalities by applying JEE technologies. It offers database operations, generate tests, determines context and language of a specific text. Therefore, the

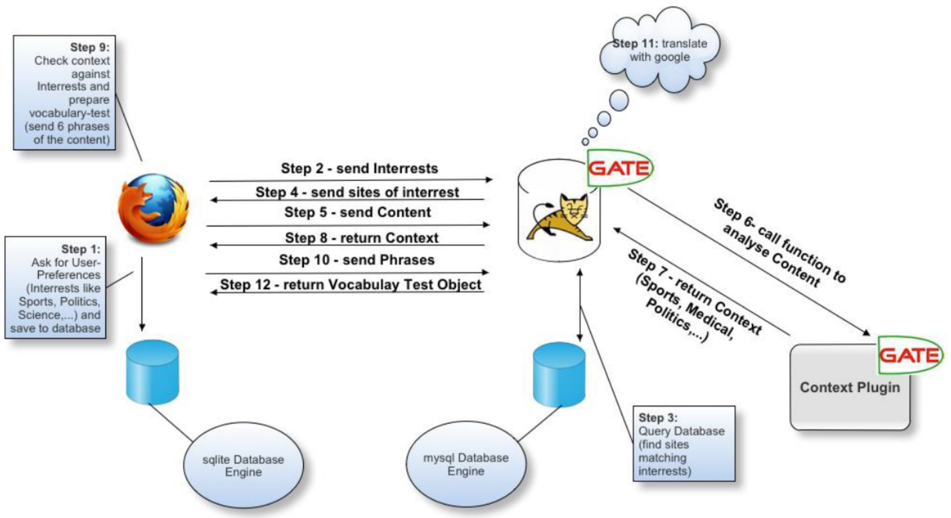

Step 9: Check context against Interests and prepare vocabulary-test (send 6 phrases of the content)
Step 1: Ask for User-Preferences (Interrests like Sports, Politics, Science,...) and save to database
Step 2 - send Interrests
Step 4 - send sites of interrest
Step 5 - send Content
Step 8 - return Context
Step 10 - send Phrases
Step 12 - return Vocabulay Test Object
Step 11: translate with google
GATE
Step 6- call function to analyse Content
Step 7 - return Context (Sports, Medical, Politics,...)
Context Plugin
Step 3: Query Database (find sites matching interests)
sqlite Database Engine
mysql Database Engine

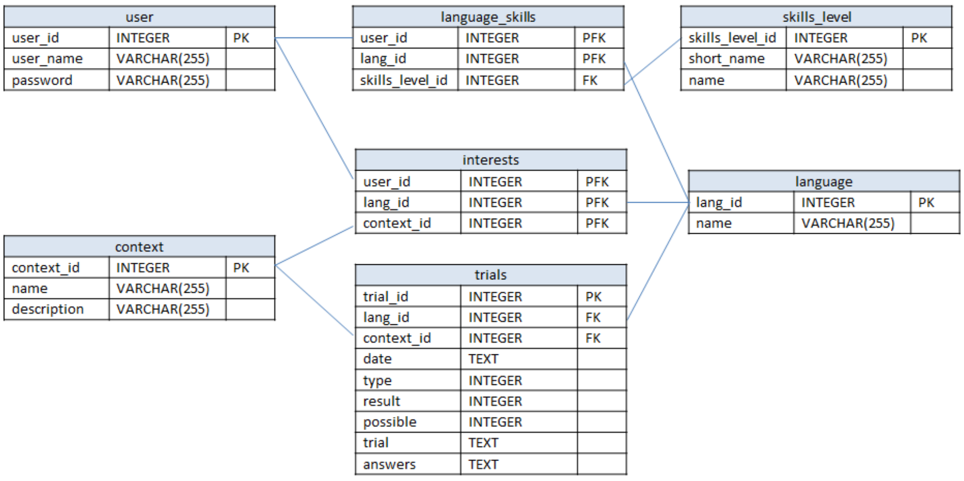
Figure 28. Technical Overview of the Initial Prototype

main purpose of the server component is to act as a kind of middleware to GATE.

The GATE Plug-in. Finally, the GATE plug-in can find out a context and the language of a text. Additionally, the plug-in checks if a match to the user preferences can be applied. Moreover, the plug-in prepares the vocabulary test based on the provided text.

Data Flows. The numbers in Figure 28 indicate the sequence of the vocabulary trainer. As precondition, a user has to define personal preferences (interests, language to be learned, etc.) which are stored in a local client database. By starting the vocabulary trainer, the local preferences are read from database ("Step 1"). Interests are sent to the server ("Step 2"), where sites are searched that match these interests ("Step 3"). By "Step 4", fitting sites are returned to the client. Then the current content of a Web site is returned to the server ("Step 5"), transferred to a GATE plug-in that analyzes the content ("Step 6") to define the fitting context ("Step 7"). The evaluated context (Medicine, IT, or Sports) is returned to the client ("Step 8"). Here the returned context is checked if it matches the personal interest and, in case, a vocabulary test is prepared by choosing a set of phrases from the content ("Step 9"). Those phrases are sent again to the server by "Step 10". In "Step 11", those phrases are translated, enriched with wrong answers as preparation for a single choice test. As final step ("Step 12"), the vocabulary test object is provided to the user.

Client Configuration and Local Database. User preferences are stored locally. Therefore, the possibility to have a local database is crucial for the vocabulary trainer client. Here the client is a Firefox plug-in, which allows the usage of SQLite, a lightweight database featuring the needs. The first step of the installation process is to check the availability of a local database. If it does not exist, it will be created. User preferences like language to learn, interest, user skills, learning level, etc. are stored here. Technically, for developers SQLite can easily be managed by an existing Mozilla Firefox add-on called “SQLite Manager”. Figure 29 shows the physical data model which indicates tables, rows, primary and foreign keys as well as corresponding data types.



user		
user_id	INTEGER	PK
user_name	VARCHAR(255)	
password	VARCHAR(255)	

language_skills		
user_id	INTEGER	PFK
lang_id	INTEGER	PFK
skills_level_id	INTEGER	FK

skills_level		
skills_level_id	INTEGER	PK
short_name	VARCHAR(255)	
name	VARCHAR(255)	

interests		
user_id	INTEGER	PFK
lang_id	INTEGER	PFK
context_id	INTEGER	PFK

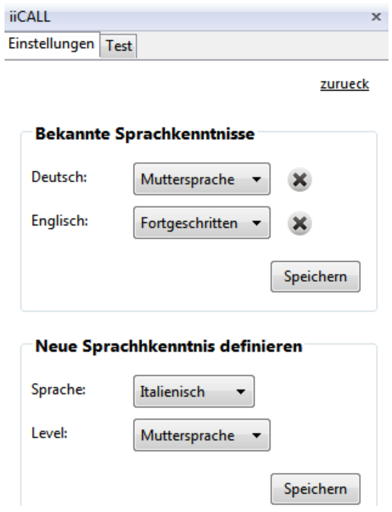
language		
lang_id	INTEGER	PK
name	VARCHAR(255)	

context		
context_id	INTEGER	PK
name	VARCHAR(255)	
description	VARCHAR(255)	

trials		
trial_id	INTEGER	PK
lang_id	INTEGER	FK
context_id	INTEGER	FK
date	TEXT	
type	INTEGER	
result	INTEGER	
possible	INTEGER	
trial	TEXT	
answers	TEXT	

FIGURE 29. Local Physical Data Model of the iiCALL Vocabulary Trainer

Once the client plug-in is successfully installed, the user is asked to define his personal language skills. Figure 30 shows the configuration window of the plug-in. Here, the user can adjust his language skills per language on the level “Beginner”, “Basic”, “Advanced”, “Native”, or “Teacher”. Currently, the vocabulary trainer allows the languages English, German, and Italian. The client plug-in uses the locale of the underlying operating system. So, here the German version of the user interface is shown.



Bekannte Sprachkenntnisse

Deutsch: Muttersprache

Englisch: Fortgeschritten

Neue Sprachhkenntnis definieren

Sprache: Italienisch

Level: Muttersprache

FIGURE 30. Configuration of Personal Language Skills

The User Interface. Graphically, the plug-in appears as a sidebar, which is located on the left side of the browser window (see Figure 31).

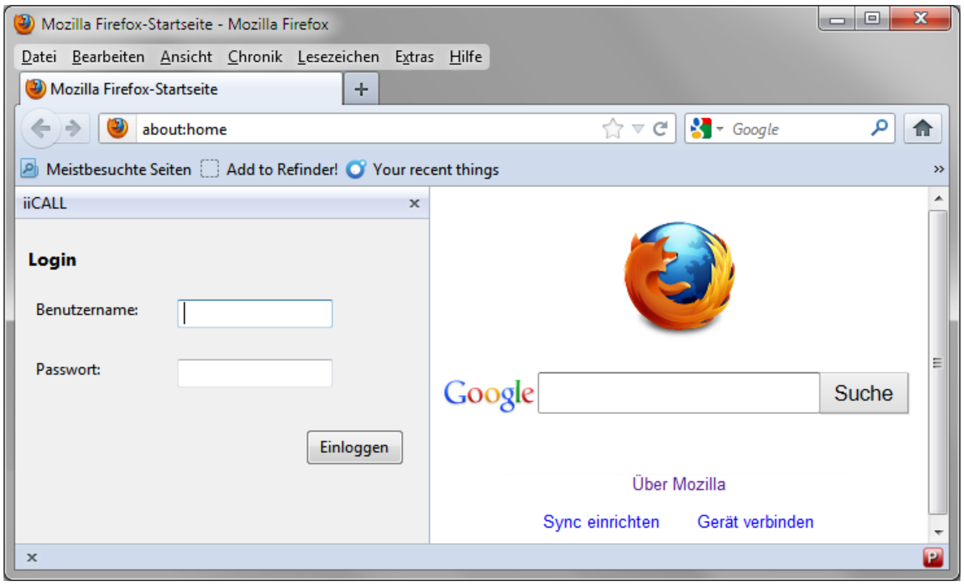
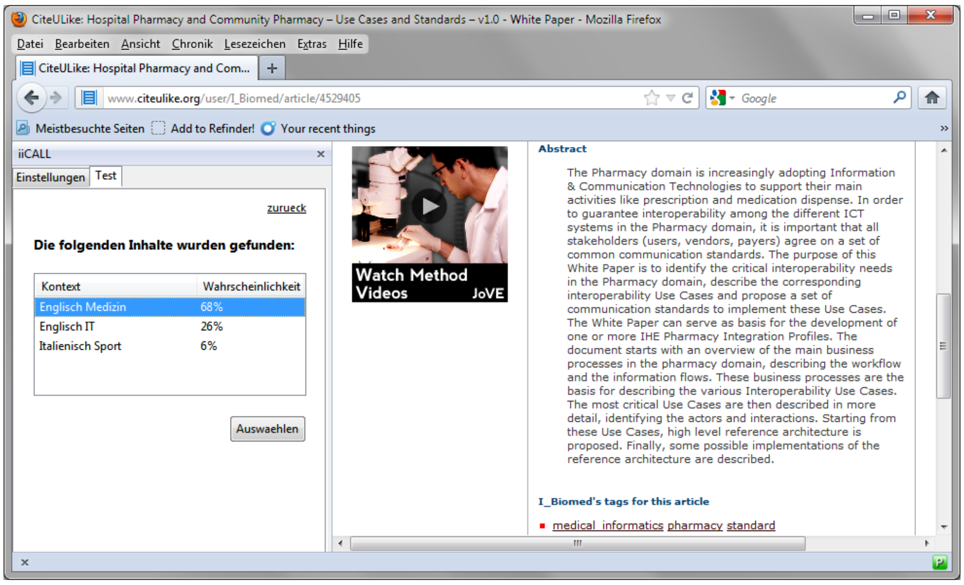


FIGURE 31. Login to the iiCALL Sidebar

The vocabulary trainer is intended to use the currently displayed content as basis text for the training. In the example given in Figure 32, the displayed text is a pharmacy white paper describing standards in hospital.


CiteULike: Hospital Pharmacy and Community Pharmacy – Use Cases and Standards – v1.0 - White Paper - Mozilla Firefox
Datei Bearbeiten Ansicht Chronik Lesezeichen Extras Hilfe
CiteULike: Hospital Pharmacy and Com...
www.citeulike.org/user/I_Biomed/article/4529405
Google
Meistbesuchte Seiten
Add to Refinder!
Your recent things
iiCALL
Einstellungen
Test
zurueck
Auswaehlen
Watch Method Videos
JoVE

Die folgenden Inhalte wurden gefunden:

Kontext	Wahrscheinlichkeit
Englisch Medizin	68%
Englisch IT	26%
Italienisch Sport	6%

Abstract

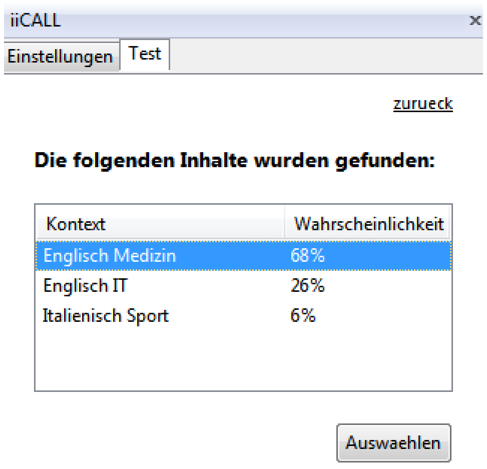
The Pharmacy domain is increasingly adopting Information & Communication Technologies to support their main activities like prescription and medication dispense. In order to guarantee interoperability among the different ICT systems in the Pharmacy domain, it is important that all stakeholders (users, vendors, payers) agree on a set of common communication standards. The purpose of this White Paper is to identify the critical interoperability needs in the Pharmacy domain, describe the corresponding interoperability Use Cases and propose a set of communication standards to implement these Use Cases. The White Paper can serve as basis for the development of one or more IHE Pharmacy Integration Profiles. The document starts with an overview of the main business processes in the pharmacy domain, describing the workflow and the information flows. These business processes are the basis for describing the various Interoperability Use Cases. The most critical Use Cases are then described in more detail, identifying the actors and interactions. Starting from these Use Cases, high level reference architecture is proposed. Finally, some possible implementations of the reference architecture are described.

I_Biomed's tags for this article

- medical_informatics pharmacy standard

FIGURE 32. Recommendation of Language and Context

Language and Context. XMLHttpRequest using SOAP Web services is the main communication technology from client to server. The content is sent to the iiCALL server. There, a GATE plug-in called ContextAnalyser gets the text to evaluate the source language and the context of it. Language identification happens mainly by statistical calculations including the number of letters and words, percentage of bigrams and trigrams. For context definition, training data in form of texts from the specific domain are used. For details of the used algorithms, we refer to [**Taus, 2013**]. The vocabulary trainer prototype supports the languages English, German, and Italian. A context to be possibly identified, must be in the area of medicine, IT, or sports. Machine learning algorithms and different thesauri are applied. The concrete example shown in Figure 32 shows the calculation returned to the client. Several possibilities (i.e. combinations of language and context) are provided to the user. Here, the combination (English, Medicine) is most probable by 68%, (English, IT) hat a probability of 26%, and even (Italian, Sports) could be possible by 6%. The user has to choose one of the provided options to continue (see Figure 33).



Die folgenden Inhalte wurden gefunden:

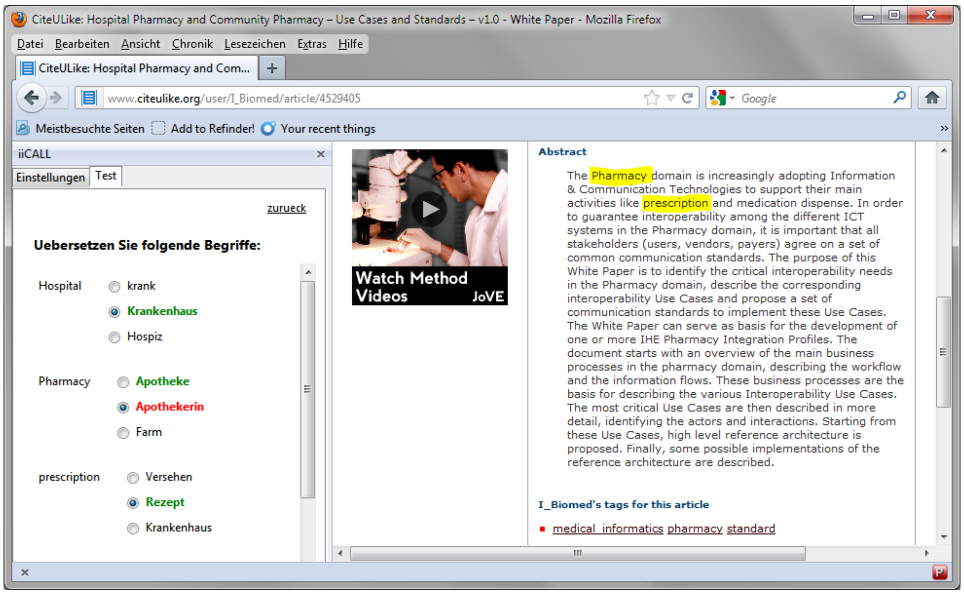
Kontext	Wahrscheinlichkeit
Englisch Medizin	68%
Englisch IT	26%
Italienisch Sport	6%

FIGURE 33. Selecting the Suggested Language and Context

The Vocabulary Test. The user must validate the provided information be selecting one of the options. The user selection is again sent back to the server (using SOAP Web service). Therefore, the WSDL interface holds the following parameters: the content text from the opened Web site, the source and destination languages, the selected context, and the number of questions for the vocabulary test [**Wahl and Winiwarter, 2012a**].

Based on these parameters a multiple choice vocabulary test (allowing only one singe answer) is created and sent back to the user interface of the client. In this example, the test consists of ten questions showing one correct and two incorrect answers each. The vocabularies themselves are taken from the text, for translations Microsoft Bing and formerly Google translation services were used.[5] In Figure 34 one can see that the words from the browser content are chosen for being used within the multiple choice test (marked in yellow).

[5]Note: license restrictions of the Google Translate API appear


CiteULike: Hospital Pharmacy and Community Pharmacy – Use Cases and Standards – v1.0 - White Paper - Mozilla Firefox
Datei Bearbeiten Ansicht Chronik Lesezeichen Extras Hilfe
CiteULike: Hospital Pharmacy and Com...
www.citeulike.org/user/I_Biomed/article/4529405
Google
Meistbesuchte Seiten
Add to Refinder!
Your recent things
iiCALL
Einstellungen
Test
zurueck
Watch Method Videos
JoVE

Uebersetzen Sie folgende Begriffe:

Hospital	krank [✓ Krankenhaus] Hospiz
Pharmacy	Apotheke [✓ Apothekerin] Farm
prescription	Versehen [✓ Rezept] Krankenhaus

Abstract

The Pharmacy domain is increasingly adopting Information & Communication Technologies to support their main activities like prescription and medication dispense. In order to guarantee interoperability among the different ICT systems in the Pharmacy domain, it is important that all stakeholders (users, vendors, payers) agree on a set of common communication standards. The purpose of this White Paper is to identify the critical interoperability needs in the Pharmacy domain, describe the corresponding interoperability Use Cases and propose a set of communication standards to implement these Use Cases. The White Paper can serve as basis for the development of one or more IHE Pharmacy Integration Profiles. The document starts with an overview of the main business processes in the pharmacy domain, describing the workflow and the information flows. These business processes are the basis for describing the various Interoperability Use Cases. The most critical Use Cases are then described in more detail, identifying the actors and interactions. Starting from these Use Cases, high level reference architecture is proposed. Finally, some possible implementations of the reference architecture are described.

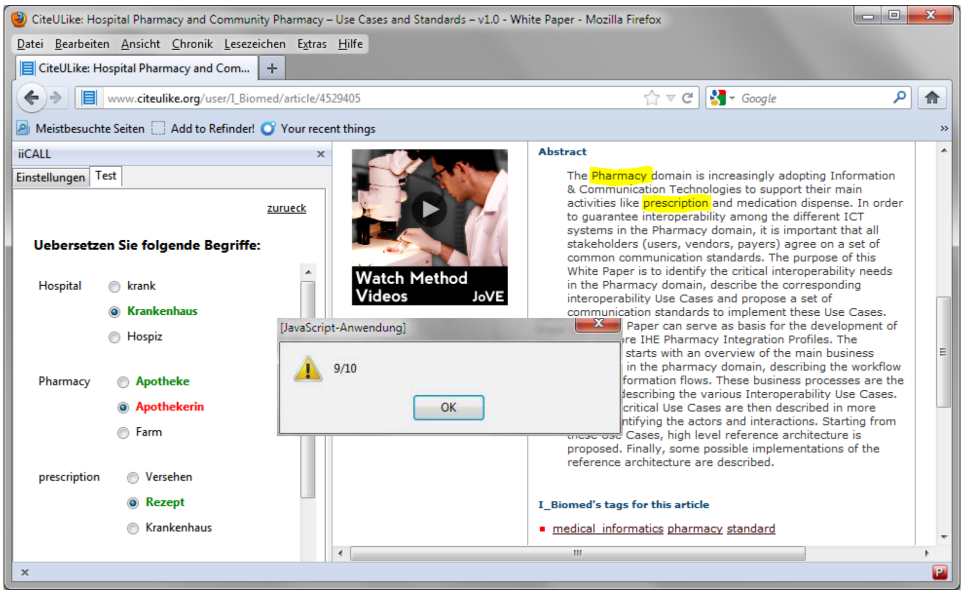
I_Biomed's tags for this article

- medical_informatics pharmacy standard

FIGURE 34. Analysis and Feedback of Multiple Choice Vocabulary Test

Test Analysis. Via SOAP Web service the number of questions are sent to the iiCALL server where a set of words is randomly selected from the browser content. For each question one word is selected. Translation into destination language is done automatically. On the one hand, the Microsoft Bing service is used, and on the other hand, translation tables are applied (e.g. for terms in specific areas like medicine). Additionally, some incorrect translations must be assigned. This happens in a way of finding "near" translations, for example, by using different forms of the translated lemma. The final test consists of the chosen word, one correct and two incorrect translations. It is transmitted back to the plug-in where it is provided to the user in form of a multiple choice test. The user tries to answer the test correctly and gets results of his test. Firstly, feedback is given in a pop-up window showing a simple statistic of the percentage of correct answers (see Figure 35). Secondly, details of correct and incorrect answers are provided in the plug-in window as well. Correct answers are marked in green and incorrect answers are marked in red (see Figure 36) [**Wahl and Winiwarter, 2012a**].

Limitations of the Initial Prototype. The very core requirements of being called an Intelligent Integrated Computer-Assisted Language Learning environment are providing language learning tasks, usage of NLP technologies, and integration in a working environment. Although the initial prototype fulfills those core requirements - it provides vocabulary training, it uses language and context determination from natural text, and it is integrated in a Web browser - it is very limited and inflexible with respect to change of functionalities and extensibility. For example, the client and the server understand only the vocabulary workflow, databases are currently limited to store user preferences, any changes need high effort in programming.



CiteULike: Hospital Pharmacy and Community Pharmacy – Use Cases and Standards – v1.0 - White Paper - Mozilla Firefox
Datei Bearbeiten Ansicht Chronik Lesezeichen Extras Hilfe
CiteULike: Hospital Pharmacy and Com...
www.citeulike.org/user/I_Biomed/article/4529405
Google
Meistbesuchte Seiten
Add to Refinder!
Your recent things
iiCALL
Einstellungen
Test
zurueck
Watch Method Videos
JoVE
[JavaScript-Anwendung]
OK

Uebersetzen Sie folgende Begriffe:

Hospital: krank [✓ Krankenhaus] Hospiz

Pharmacy: Apotheke [✓ Apothekerin] Farm

prescription: Versehen [✓ Rezept] Krankenhaus

9/10

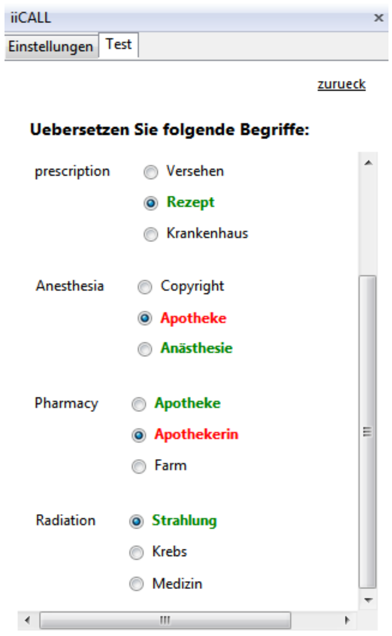
Abstract

The Pharmacy domain is increasingly adopting Information & Communication Technologies to support their main activities like prescription and medication dispense. In order to guarantee interoperability among the different ICT systems in the Pharmacy domain, it is important that all stakeholders (users, vendors, payers) agree on a set of common communication standards. The purpose of this White Paper is to identify the critical interoperability needs in the Pharmacy domain, describe the corresponding interoperability Use Cases and propose a set of communication standards to implement these Use Cases. Paper can serve as basis for the development of re IHE Pharmacy Integration Profiles. The starts with an overview of the main business in the pharmacy domain, describing the workflow formation flows. These business processes are the describing the various Interoperability Use Cases. critical Use Cases are then described in more ntifying the actors and interactions. Starting from these Use Cases, high level reference architecture is proposed. Finally, some possible implementations of the reference architecture are described.

I_Biomed's tags for this article

- medical_informatics pharmacy standard

FIGURE 35. Feedback of Result of Multiple Choice Vocabulary Test



Uebersetzen Sie folgende Begriffe:

prescription: Versehen [✓ Rezept] Krankenhaus

Anesthesia: Copyright [✓ Apotheke] Anästhesie

Pharmacy: Apotheke [✓ Apothekerin] Farm

Radiation: [✓ Strahlung] Krebs Medizin

FIGURE 36. Analysis of Test

CHAPTER 8

The iiCALL Development Framework Version 1

The context-related vocabulary trainer was our first try to implement language learning tasks within a Web browser. Actually, it did not really follow the idea of the draft iiCALL Development process (see Figure 25). So, at this stage of development, the system was very inflexible. Process steps were more or less hard-coded, a new functionality was difficult to add. There was no common semantic understanding of transmitted data. We had to introduce a new development perspective.

Based on the HL7 HDF and the underlying HL7 V3 Reference Information Model Backbone (see Figure 20 on page 48), and under refinement of the iiCALL Development Process Draft Version (see Figure 25 on page 54) and the core classes of iiCALL GDM Draft Version (see Figure 24 on page 53), we defined the iiCALL GDM-Core Version 1 combined with the iiCALL Development Process Version 1. Here, we introduce both as basis for the implementation of the iiCALL Software Framework.

8.1. The iiCALL Core Generic Data Model in Version 1

The refined data model for the iiCALL environment essentially corresponds to the iiCALL GDM Draft Version (see Figure 24 on page 53). An entity (Entity) can participate in an action (Act) within a specific role (Role) in a certain way (Participation). Individual actions can be concatenated with one another (Act Relationship) to define learning sequences. The Draft Version of the GDM has two additional elements, which are a state (State) linked to an entity to map a state to the entity, for example, the skill levels of learners. Secondly, the connector (Connector) joins an entity with an act to reference external standards like LMF. Version 1 of the GDM-Core removes those both elements, since this information can also be modeled alternatively. So, the Version 1 of the GDM consists of five elements, as shown in Figure 37 [**Wahl et al., 2015b, Wahl et al., 2015a, Galler, 2015**].

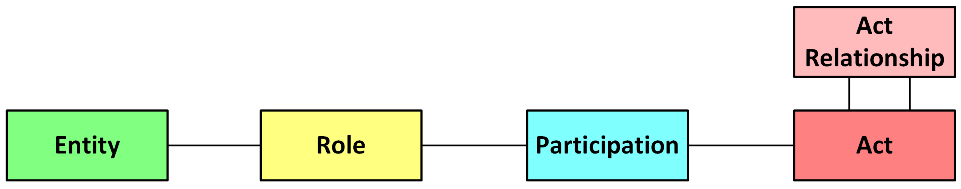


FIGURE 37. The iiCALL GDM-Core in Version 1

8.2. The iiCALL Development Process in Version 1

According to the HL7 methodology and the Draft Version of the iiCALL Development Process (see Figure 25 on page 54), we introduce an adopted and more adequate development process. It consists of three phases, which are concept, design and implementation (see Figure 38) [**Galler, 2015, Wahl et al., 2015b, Wahl et al., 2015a**].

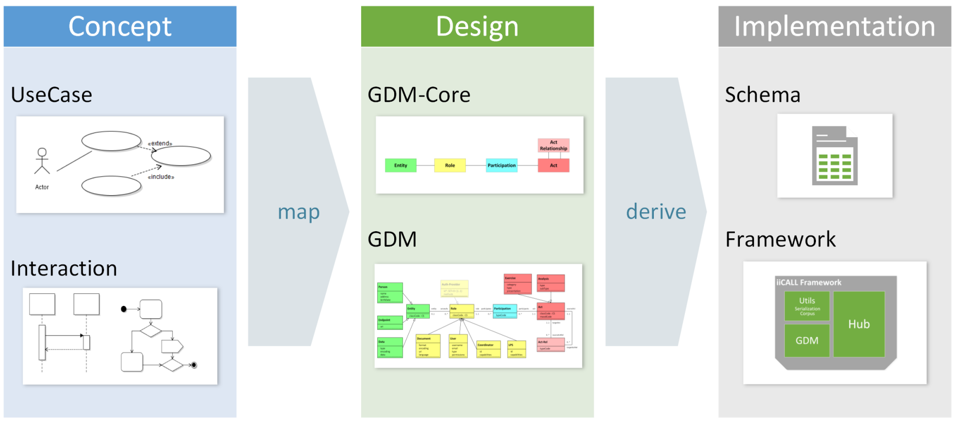


FIGURE 38. The iiCALL Development Process in Version 1

Concept. The concept phase starts with the requirements engineering, i.e. the analysis of the application and the associated requirements of the learning environment. These can be captured verbally or textually using UML use case notations (comparable to the storyboard concept of the HL7). Afterward, UML sequence diagrams are used to describe interactions between the involved systems. In addition, behavior of individual systems can be declared using UML activity diagrams, which will result in message sending actions (comparable to the HL7 trigger events), or sequence diagrams.

Design. During design phase, the use cases are mapped to the data model based on their descriptions. The focus lies on the information exchanged between the individual systems. For this purpose, they are initially assigned to the elements of the GDM-Core and from this, those core elements are expanded by specialization by adding additional specific attributes. For the interoperability of the systems, it is essential that the structure of the GDM-Core remains stable. In contrary, the GDM can be additionally adapted, refined or restricted. This process corresponds to the HL7 R-MIM, and D-MIM respectively.

Implementation. Implementation is done on a basis of individual systems. Therefore, an XML schema definition is specified from the refined GDM. This schema declares the structure of all classes of the GDM, including their attributes and allowed values. On the basis of such schemata, different software frameworks in different technologies can be developed. These individual frameworks support developers of the individual systems in the realization of their specific applications. Thus, interoperability between systems based on the GDM can be guaranteed.

CHAPTER 9

Requirements Engineering

In the following sections we provide use cases and their mappings to the R-MIM. The use cases give a comprehensive specification of the functionalities of an iiCALL system.

9.1. Use Cases for a Core iiCALL System

Several use cases are modeled using UML use case diagrams combined with use case specifications. They are specified verbally indicating pre- and post-conditions, basic and alternate paths, and possible exceptions. These use cases are "Login" (see Figure 39), "Grading Test" (see Figure 40), "Do Test" (see Figure 41), "Chat" (see Figure 42), "Ask for Help" (see Figure 43), "Sync Data" (see Figure 44), "User Data" (see Figure 45), and "Workgroup" (see Figure 46).

Use Case Diagram Login. The use cases collected in the diagram "Login" deal with registration and login purposes. For Facebook login, the Facebook service must be involved. The iiCALL service mainly offers validity checks (see Figure 39).

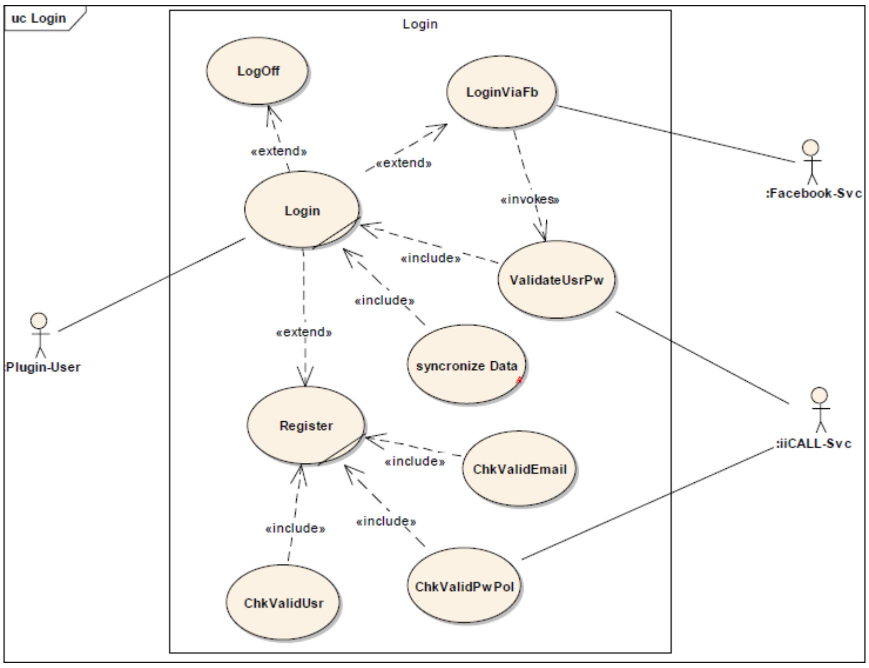


Figure 39. Use Case Diagram "Login"

Use Case Login. A user wants to login to the iiCALL system. Either the user registers directly using the iiCALL login mechanism, or alternatively, a login using a Facebook account is possible. A list of available login mechanisms is displayed in case the user has not provided preregistered login credentials (see Table 3).

Table 3. Use Case Login

Pre-condition	not relevant
Basic path	Login happens via the integrated iiCALL login mechanism using preregistered login credentials.
Alternate path	Login is done using a Facebook account.
Exceptions	Login is not possible due to iiCALL system unavailability, the Facebook login mechanism is not responding, or the Facebook service is currently offline. The user is informed that login is not possible and should be tried later. User is not logged in.
Post-condition	User is logged in.

Use Case LoginViaFb. A user wants to login to the iiCALL system by using an existing Facebook account (see Table 4).

Table 4. Use Case LoginViaFb

Pre-condition	User needs to have a valid Facebook account.
Basic path	User uses a Facebook login and the Facebook service returns the validity of the provided credentials.
Alternate path	User uses a Facebook login and the Facebook service returns that the credentials are not correct. User cannot login.
Exceptions	The Facebook login mechanism is not working or offline. The user gets informed that the Facebook service is currently not available and should be tried later. Alternatively, the user could try the integrated iiCALL login mechanism. User is not logged in.
Post-condition	User is logged in.

Use Case Register. User is not registered within the iiCALL system and wants to use the iiCALL integrated login mechanism. The user provides personal data which have to be re-checked by the iiCALL system (see Table 5).

TABLE 5. Use Case Register

Pre-condition	The user is not registered.
Basic path	The user defines mandatory data account name, password, and email address to be identified. Complementary data are combinations of language and corresponding skill levels, the user's learning preferences, i.e. language and context the user wants to learn.
Alternate path	If a user decides to use a Facebook account for login, the user can add the Facebook account name and complementary data as mentioned in the basic path above.
Exceptions	The registration service is not available. The user gets informed that the service is down and registration service will be available later.
Post-condition	A confirmation link is sent to the user and confirmation is awaited.

Use Case ChkValidEmail. This use case is part of the registration process. It checks the validity of a provided email address (see Table 6).

TABLE 6. Use Case ChkValidEmail

Pre-condition	The user has started to register for the iiCALL system.
Basic path	The iiCALL query service for duplicate email addresses returns that the provided email address is valid in case the email address is not already taken by another user.
Alternate path	The provided email address is not valid or already taken by another user. The user is informed to restart registration process.
Exceptions	The iiCALL service to check email validity is not available. The user gets informed that the service is down and registration service will be available later. The check of provided email address cannot be fulfilled.
Post-condition	The check of provided email address is positive, and a confirmation is returned.

Use Case ChkValidPwPol. This use case is part of the registration process. It downloads password policy from server and performs password validity check. (see Table 7).

TABLE 7. Use Case ChkValidPwPol

Pre-condition	The user has started to register for the iiCALL system.
Basic path	Password policy is downloaded from iiCALL service and the password fits policy requirements. The service returns that password is valid.
Alternate path	The provided password does not fit the password policy. The user gets informed to choose a different password.
Exceptions	The iiCALL service is not available or the policy cannot be downloaded. The user gets informed that the service is down and registration service will be available later. The check of provided password cannot be fulfilled.
Post-condition	The check of provided password is positive, and a confirmation is returned.

Use Case ChkValidUsr. This use case is part of the registration process. It checks the validity of a provided username. User has entered a username to use within the iiCALL System, which verifies if the username is already taken. (see Table 8).

TABLE 8. Use Case ChkValidUsr

Pre-condition	The user has started to register for the iiCALL system.
Basic path	The existence of provided username is checked. Username does not exist in iiCALL system database. The service returns that username is valid.
Alternate path	The provided username does already exist. The user gets informed to choose a different username.
Exceptions	The iiCALL service is not available. The user gets informed that the service is down and service will be available later. The check of provided username cannot be fulfilled.
Post-condition	The check of provided username is positive, and a confirmation is returned.

Use Case ValidateUsrPw. This use case is part of the login process. Validation of username and password combination happens within the iiCALL system (see Table 9).

TABLE 9. Use Case ValidateUsrPw

Pre-condition	The user has tried to login to the iiCALL system using a username and password combination.
Basic path	The provided username and password combination is validated. The service returns that username and password combination is valid.
Alternate path	The provided username/password combination is not valid. The user gets informed that username and password combination is not valid.
Exceptions	The iiCALL service is not available. The user gets informed that the service is down and service will be available later. The validation of provided username and password combination cannot be fulfilled.
Post-condition	The check of provided username and password combination is positive, and a confirmation is returned. User is allowed to login.

Use Case LogOff. This use case performs logoff from the iiCALL system (see Table 10).

TABLE 10. Use Case LogOff

Pre-condition	User is logged in and wants to logoff.
Basic path	User logged off successfully.
Alternate path	User is inactive for a predefined time. User gets logged off automatically.
Exceptions	User cannot perform a clean logoff because iiCALL service is not available.
Post-condition	User is logged off and session parameters are invalidated.

Use Case Diagram Grading Test. The use cases collected in the diagram "Grading Test" implement initial grading tests to get the level of users (see Figure 40).

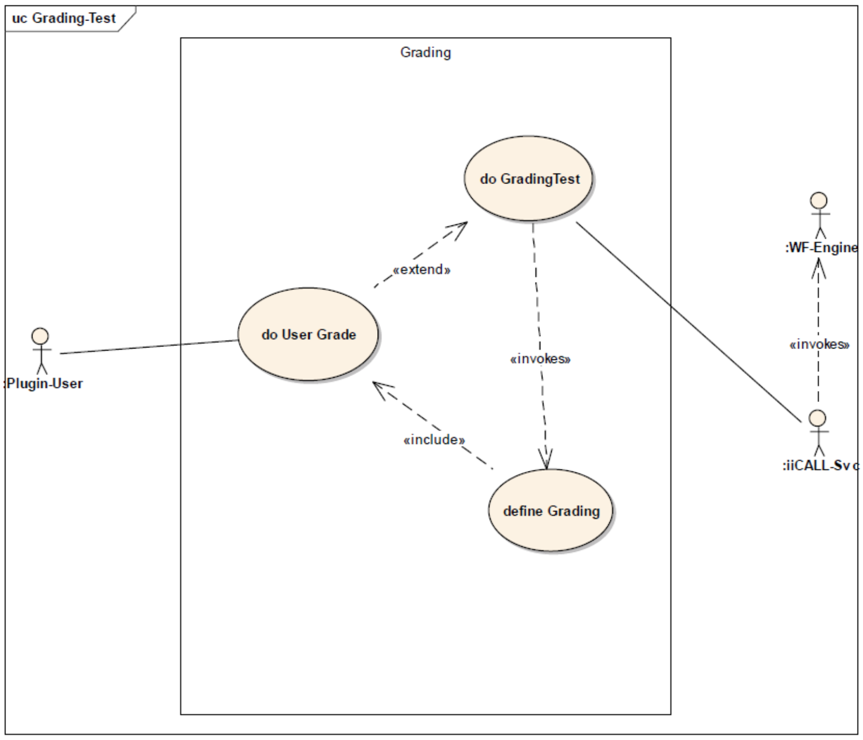


FIGURE 40. Use Case Diagram "Grading Test"

Use Case Do User Grade. User wants to set his grading within the iiCALL Systems. He either requests a grading test or simply adds a grading based on a best guess method. The following levels should be defined within the system: “Beginner”, “Basic”, “Advanced”, “Native”, or “Teacher” (see Table 11).

Table 11. Use Case Do User Grade

Pre-condition	User has no level assigned for a specific language or wants to re-do the assignment for a specific language.
Basic path	The user asks for a grading test for a specific language and runs the test to get assigned corresponding predefined levels.
Alternate path	The user is able to run a self-assessment. Therefore, the user can choose combinations of languages and corresponding predefined levels (“Beginner”, “Basic”, “Advanced”, “Native”, or “Teacher”).
Exceptions	The iiCALL service for grading test is not available. The user gets informed that the service is down and service will be available later. The assignment of predefined level for a specific language cannot be fulfilled.
Post-condition	One predefined level for a specific language is initially assigned to the user.

Use Case Define Grading. User does not want to do a grading test and defines a rating on a best-guess basis. It might be possible to downgrade or upgrade the user skills based on completed tests during the usage of the system. (see Table 12).

Table 12. Use Case Define Grading

Pre-condition	The user has started the grading process for a specific language using the iiCALL system
Basic path	The user asks for a grading test for a specific language and runs the test.
Alternate path	not relevant
Exceptions	User cannot run the grading test because iiCALL service is not available or test is not finished. A predefined level for a specific language cannot be assigned.
Post-condition	One predefined level for a specific language is assigned to the user.

Use Case Do GradingTest. User requests a grading test for a specific language from the iiCALL system. (see Table 13).

TABLE 13. Use Case Do GradingTest

Pre-condition	The user has started the grading process for a specific language using the iiCALL system
Basic path	A grading test is generated and provided to the user.
Alternate path	not relevant
Exceptions	User cannot perform the grading because iiCALL service is not available.
Post-condition	One predefined level for a specific language is initially assigned to the user.

Use Case Diagram Do Test. The use cases shown in the diagram "Do Test" collect all use cases related to specific language learning tests, which are, for instance, vocabulary tests, cloze tests, etc. The kind of test provided by the iiCALL system depends on user configuration as well as on available learning scenarios for the specific language and content. Users can pause and resume tests to deepen their already gained language skills. Test results can be used to re-grade the user level (see Figure 41).

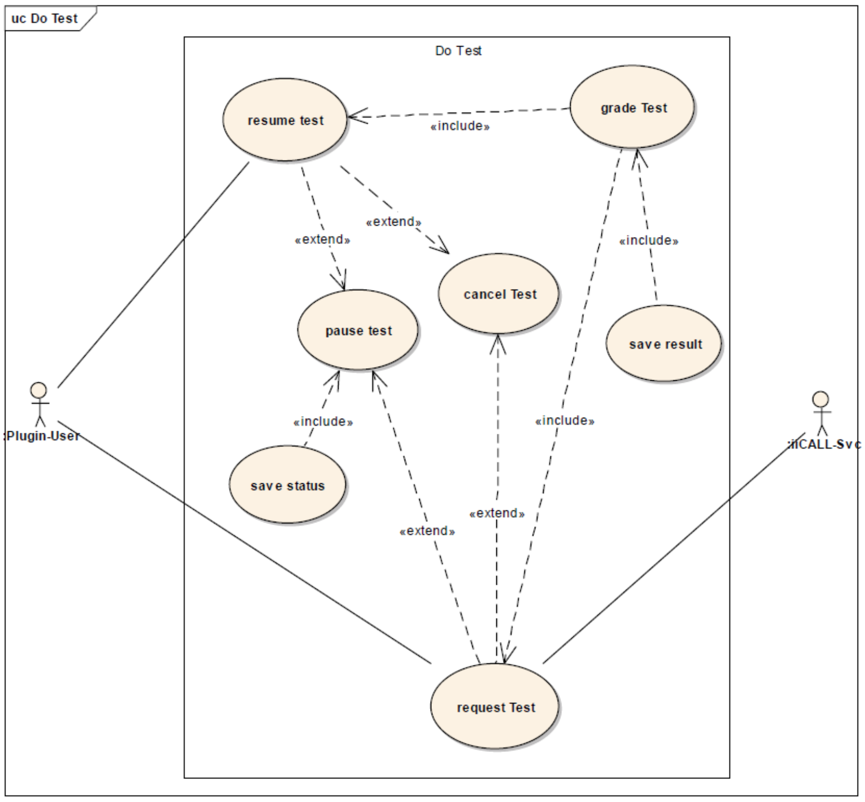


FIGURE 41. Use Case Diagram "Do Test"

Use Case Request Test. The user applies for a test for language learning. Differences occur depending on learning preferences (see Table 14).

TABLE 14. Use Case Request Test

Pre-condition	The user wants to use the iiCALL system to get tests related to specific languages.
Basic path	The user has configured specific preferences. Due to these preferences (e.g. language, relevant context, or language skill level) the iiCALL system provides a proper test.
Alternate path	The user has not configured any specific preferences, so the iiCALL system offers a choice of available tests (vocabulary test, cloze test, etc.) for the specific language. The user has to choose a specific test from the list.
Exceptions	The iiCALL service is not available. The user gets informed that the service is down and service will be available later. Alternatively, no proper test is available for language to be learned and/or relevant context. A proper test cannot be provided to the user.
Post-condition	A proper test is provided to the user.

Use Case Grade Test. A specific test gets evaluated by the iiCALL system. (see Table 15).

TABLE 15. Use Case Grade Test

Pre-condition	A test has been fully finished by the user.
Basic path	Test is finished and results are evaluated (graded) by the iiCALL system. The user gets informed about the result which is internally stored and saved following the use case "Save Result" (see Table 16)
Alternate path	not relevant
Exceptions	Test cannot be evaluated because the iiCALL evaluation service is not available. User gets informed. Test is not evaluated (graded).
Post-condition	A test is evaluated (graded).

Use Case Save Result. Test results can be stored to be reviewed later (see Table 16).

Table 16. Use Case Save Result

Pre-condition	User has finished a test and the iiCALL service has evaluated (graded) the test
Basic path	Specific meta-data of the test itself (kind of test, all test steps and user responses, the start and finishing timestamps) are collected and stored using the iiCALL storage service
Alternate path	not relevant
Exceptions	Test cannot be stored because the iiCALL storage service is not available. User gets informed. Result is not stored.
Post-condition	A test result is internally saved with specific meta-data of the test itself (kind of test, all test steps and user responses, the start and finishing timestamps).

Use Case Cancel Test. An ongoing test is canceled and will not be continued later. (see Table 17).

Table 17. Use Case Cancel Test

Pre-condition	User runs a test.
Basic path	The user wants to stop the test and cancels it
Alternate path	not relevant
Exceptions	not relevant
Post-condition	Test is canceled. Results are not evaluated (graded) and nothing is stored.

Use Case Pause Test. The user wants to interrupt the current test with the goal to resume the test later. (see Table 18).

TABLE 18. Use Case Pause Test

Pre-condition	User runs a test and wants to interrupt the test and stop it with the intention to resume the test later.
Basic path	The system evaluates the finished part of test. The user gets informed about the result (test status) which is internally stored and saved following the use case “Save Status” (see Table 19)
Alternate path	not relevant
Exceptions	Test cannot be evaluated because the iiCALL evaluation service is not available. User gets informed. Test status is not evaluated.
Post-condition	A test status is evaluated.

Use Case Save Status. Test status, i.e results of the so far finished test, can be stored to be resumed later. (see Table 19).

TABLE 19. Use Case Save Status

Pre-condition	User has paused a test and the iiCALL service has evaluated the test status.
Basic path	Specific meta-data of the so far finished test (kind of test, all test steps and user responses, the start and pausing timestamps) are collected and stored using the iiCALL storage service
Alternate path	not relevant
Exceptions	Test status cannot be stored because the iiCALL storage service is not available. User gets informed. Status is not stored.
Post-condition	A test status is internally saved with specific meta-data of the test (kind of test, all test steps and user responses, the start and pausing timestamps).

Use Case Resume Test. A user can resume from a previous paused (and therefore stored) test. (see Table 20).

Table 20. Use Case Resume Test

Pre-condition	User has paused a test and the iiCALL storage service has stored the test status.
Basic path	User chooses from a list of already paused tests the specific test to be resumed. The corresponding test status is loaded and the test can be resumed starting at the previous last test step.
Alternate path	not relevant
Exceptions	The iiCALL storage service is not available or a test status cannot be loaded. User gets informed. Test cannot be resumed.
Post-condition	Test is ready to be resumed.

Use Case Diagram Chat. The use cases collected in the diagram "Chat" address chat-related functionalities. Chats are meant for virtual discussions or communications for collaborative learning tasks (see Figure 42).

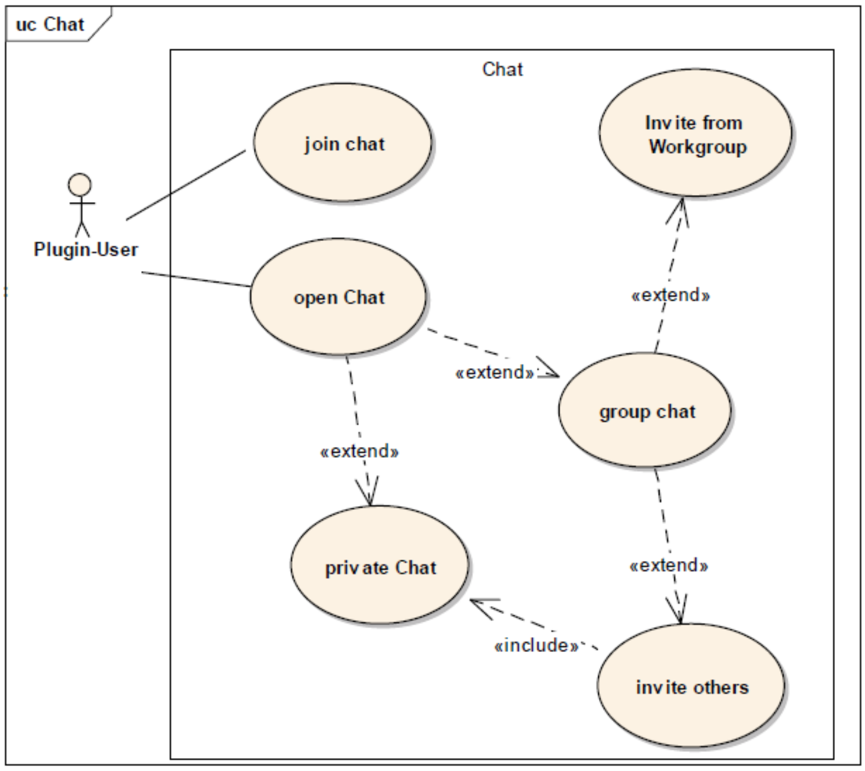


Figure 42. Use Case Diagram "Chat"

Use Case Open Chat. The user creates a chat environment which can either be a private chat or a group chat. Private chats are only for invited participants, whereas group chats allow any user to join (see Table 21).

Table 21. Use Case Open Chat

Pre-condition	The user wants to virtually discuss with others in a chat environment.
Basic path	The user defines a name for the chat, which is checked for uniqueness by the iiCALL chat service. If the chosen name is not unique, the iiCALL chat service asks for a different name. Optionally, the user can add a content description to specify the learning topic. The use case "Private Chat" follows (see Table 22)
Alternate path	The user wants to create a group chat. Definition of unique name is needed, and group chats additionally need a mandatory description to be searchable for other members. Here, the use case "Group Chat" follows (see Table 23)
Exceptions	The iiCALL chat service is not available. User gets informed to try it later. Chat environment cannot be created.
Post-condition	A chat environment with a unique name and an optional content description is created.

Use Case Private Chat. In private chats only invited attendees are allowed to join. (see Table 22).

Table 22. Use Case Private Chat

Pre-condition	The user has successfully created a chat environment (see use case "Open Chat" in Table 21)
Basic path	The user who has created the chat environment, marks this environment to be treated as "private chat". The user invites other users to join. For each invitation, the use case "Invite Others" is relevant (see Table 24)
Alternate path	not relevant
Exceptions	The iiCALL chat service is not available. User gets informed to try it later. Private chat cannot be started.
Post-condition	Private chat is successfully started.

Use Case Group Chat. In group chats invited attendees are allowed to join and, additionally, all interested users of the iiCALL system may join. (see Table 23).

TABLE 23. Use Case Group Chat

Pre-condition	The user has successfully created a chat environment (see use case "Open Chat" in Table 21)
Basic path	The user, who has created the chat environment, marks this environment to be treated as "group chat". The user invites other users to join. For each invitation, the use case "Invite Others" is relevant (see Table 24).
Alternate path	Users, who are interested in joining, can participate following the use case "Join Chat" (see Table 26).
Exceptions	The iiCALL chat service is not available. User gets informed to try it later. Group chat cannot be started.
Post-condition	Group chat is successfully started.

Use Case Invite Others. A user who has successfully created a chat environment, be it a group chat er a private chat, can invite iiCALL users to join. (see Table 24).

TABLE 24. Use Case Invite Others

Pre-condition	A chat environment (private or group) is successfully started.
Basic path	The user who has created the chat environment, chooses other users to be invited. These other users can be identified by a search for user name or email address. The user can optionally add an invitation message. All other users get a chat invitation with an optional test message.
Alternate path	not relevant
Exceptions	Ohter users cannot be found. No invitations are sent.
Post-condition	All other users have been invited to join the chat.

Use Case Invite from Workgroup. Workgroups are members who collaboratively work within a specific learning task. All users who belong to a workgroup are invited to join a group chat (see Table 25).

Table 25. Use Case Invite from Workgroup

Pre-condition	A group chat is successfully started (see use case "Group Chat" in Table 23). A workgroup has already been created (see use case "New Workgroup" in Table 48).
Basic path	The user who has created the group chat environment, chooses a workgroup to be invited. The workgroup can be identified by a unique workgroup name or by the search within the optional workgroup description. The user can optionally add an invitation message. All members of the workgroup get a chat invitation with an optional test message.
Alternate path	not relevant
Exceptions	Workgroup cannot be found. No invitations are sent.
Post-condition	All members of the selected workgroup have been invited to join the chat.

Use Case Join Chat. Joining a chat can happen as reaction to an invitation. Without invitation, users can join only group chats, which they can find by a search for interesting key words given in the chat descriptions (see Table 26).

Table 26. Use Case Join Chat

Pre-condition	A chat environment (private or group) is successfully started.
Basic path	User has gotten an invitation as result of use cases "Invite Others" (see Table 24) or "Invite from Workgroup" (see Table 25). By answering the invitation, the user can join the chat where the invitation comes from.
Alternate path	User finds a chat by searching for a specific keyword within all available chat descriptions.
Exceptions	Here, two cases are relevant. Firstly, the iiCALL chat service is not available. User gets informed to try later. Secondly, a chat cannot be found by the search for keyword. User cannot join the chat.
Post-condition	User has successfully joined the chat.

Use Case Diagram Ask for Help. The use cases in the diagram "Ask for Help" combine all functionalities for users to ask specific language learning questions, either to the iiCALL community (which are all iiCALL users) or to a social network (see Figure 43).

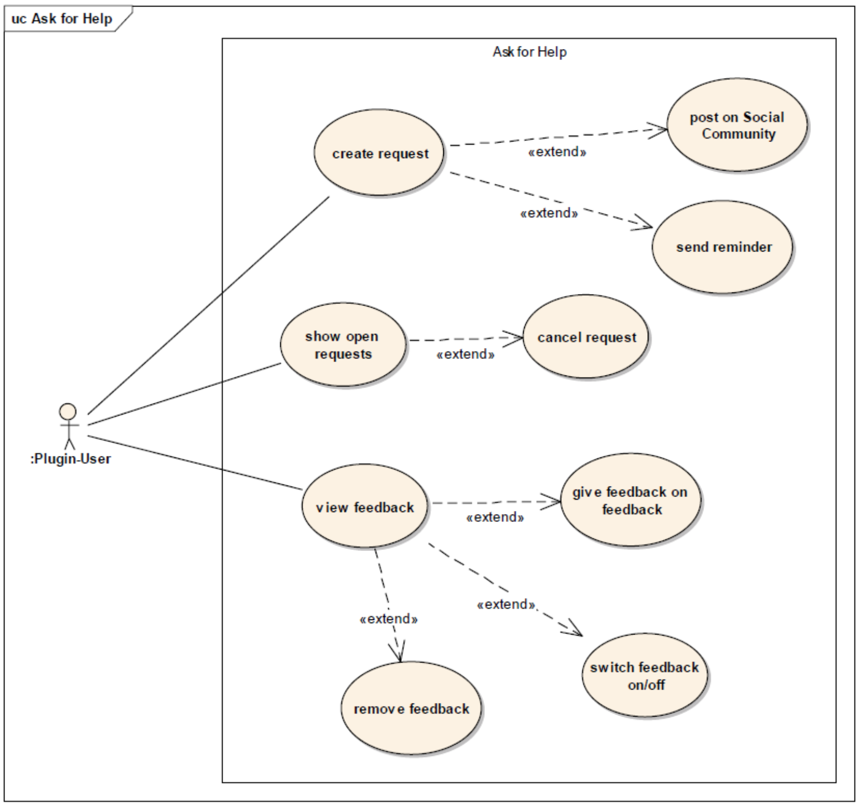


FIGURE 43. Use Case Diagram "Ask for Help"

Use Case Create Request. The user posts a question to the iiCALL community (see Table 27).

Table 27. Use Case Create Request

Pre-condition	The user needs help from the language learning community
Basic path	For a specific request, the user defines language, the learning topic, and the question text and posts it to the iiCALL system.
Alternate path	Alternatively, the user can post the question to a social network (see use case "Post on Social Community" in Table 28). Therefore, account data must be pre-configured.
Exceptions	The iiCALL question service is not available. User gets informed to try it later. Question cannot be posted.
Post-condition	Question is posted with status "open".

Use Case Post on Social Community. In addition to posting questions to the iiCALL system, the user can forward the question to a social community (see Table 28).

Table 28. Use Case Post on Social Community

Pre-condition	User has posted a question through the iiCALL question service (see use case "Create Request" in Table 27)
Basic path	The user chooses a specific social community from a list of available social communities where questions can be posted.
Alternate path	not relevant
Exceptions	Account data for the selected social network are not configured correctly, the social network returns an error message. The user is informed to update the pre-configured account data for the specific social network. Question is not posted on the social media.
Post-condition	Question is posted on the social media.

Use Case Send Reminder. Sending a reminder means that a formerly posted question can be edited and re-posted to the iiCALL question service. So, by the reminder the question gets updated and will be re-posted. Relevant users get informed (see Table 29).

Table 29. Use Case Send Reminder

Pre-condition	The user has posted a question.
Basic path	The user chooses a specific question from the list of open questions which he posted formerly (see use case "Show Open Requests" in Table 30 filtering his personal username). The user has the option to re-edit the question before he re-posts it. Relevant users, i.e. users who are within the same learning topic, get informed. In the list of questions the re-posted question will be ranked on top position.
Alternate path	not relevant
Exceptions	The iiCALL question service is not available. User gets informed to try it later. Question cannot be selected.
Post-condition	Reminder is sent, the status of the question is set to "open".

Use Case Show Open Requests. The user can search through open questions where the search criteria are users, learning topics, or keywords. (see Table 30).

Table 30. Use Case Show Open Requests

Pre-condition	There exist open questions in the iiCALL system.
Basic path	The user can ask for open requests filtering learning topics, keywords, or user names. Keywords are used to perfom full-text search within the question text.
Alternate path	not relevant
Exceptions	The iiCALL question service is not available. User gets informed to try it later. Open requests cannot be shown.
Post-condition	The list of open requests is presented to the user.

Use Case Cancel Request. A user can cancel one of his personal open questions (see Table 31).

Table 31. Use Case Cancel Request

Pre-condition	The user has posted a question.
Basic path	The user chooses a specific question from the list of open questions which he posted formerly (see use case “Show Open Requests” in Table 30 filtering his personal username). By canceling, the question status is switched from “open” to “canceled”. The question remains visible for the user, and the question becomes invisible for all others.
Alternate path	not relevant
Exceptions	The iiCALL question service is not available. User gets informed to try it later. Question cannot be canceled.
Post-condition	The question is canceled.

Use Case View Feedback. The user can browse through feedback to the questions he has posted formerly (see Table 32).

Table 32. Use Case View Feedback

Pre-condition	The user has posted a question.
Basic path	By selecting one of own questions, the user gets presented the list of feedbacks to browse through.
Alternate path	The user can filter the list of questions by username or dates.
Exceptions	The iiCALL question service is not available. User gets informed to try it later. Feedback cannot be presented.
Post-condition	Feedback list is provided.

Use Case Give Feedback on Feedback. For a provided feedback, the user can respond. The response is stored in the feedback tree and the corresponding user gets informed (see Table 33).

TABLE 33. Use Case Give Feedback on Feedback

Pre-condition	The user has posted a question.
Basic path	The user chooses a specific question from the list of questions which he posted formerly. The user views feedback to the selected question (see use case "View Feedback" in Table 32). Then, a specific feedback is selected and re-answered.
Alternate path	The user can filter the list of questions by username or dates.
Exceptions	The iiCALL question service is not available. User gets informed to try it later. Feedback cannot be responded.
Post-condition	The corresponding user gets informed that there exist responses to his feedback.

Use Case Remove Feedback. The user can select specific feedback given and removes it. So, the feedback is not visible any more (see Table 34).

TABLE 34. Use Case Remove Feedback

Pre-condition	The user has posted a question to which at least one feedback is given.
Basic path	The user chooses a specific question from the list of open questions which he posted formerly (see use case "Show Open Requests" in Table 30 filtering his personal username). The user views feedback to the selected question (see use case "View Feedback" in Table 32) and removes it. Optionally, the user can add a note that indicates the reason.
Alternate path	not relevant
Exceptions	The iiCALL question service is not available. User gets informed to try it later. Feedback cannot be removed.
Post-condition	Feedback is removed and becomes invisible for other users.

Use Case Switch Feedback on/off. Switching feedback "off" means that an open question is set inactive. Users cannot respond to a posted question any more. By switching feedback "on" again, responding is possible again (see Table 35).

TABLE 35. Use Case Switch Feedback on/off

Pre-condition	The user has posted a question.
Basic path	The user chooses a specific question from the list of open questions which he posted formerly (see use case "Show Open Requests" in Table 30 filtering his personal username). Then, the user revokes the right of giving feedback to the question.
Alternate path	A question which currently cannot be answered, the right of giving feedback to it is re-assigned.
Exceptions	The iiCALL question service is not available. User gets informed to try it later. Feedback cannot be switched on/off.
Post-condition	The right of giving feedback to a specific question is changed.

Use Case Diagram Sync Data. The use cases collected in the diagram "Sync Data" deal with synchronizing user-specific data between iiCALL server and the client. When local modifications are identified, the user has the option to synchronize them to the iiCALL database or discard all local changes. Changes on the server side will always be synchronized to the client. In this case, the user will not be asked (see Figure 44).

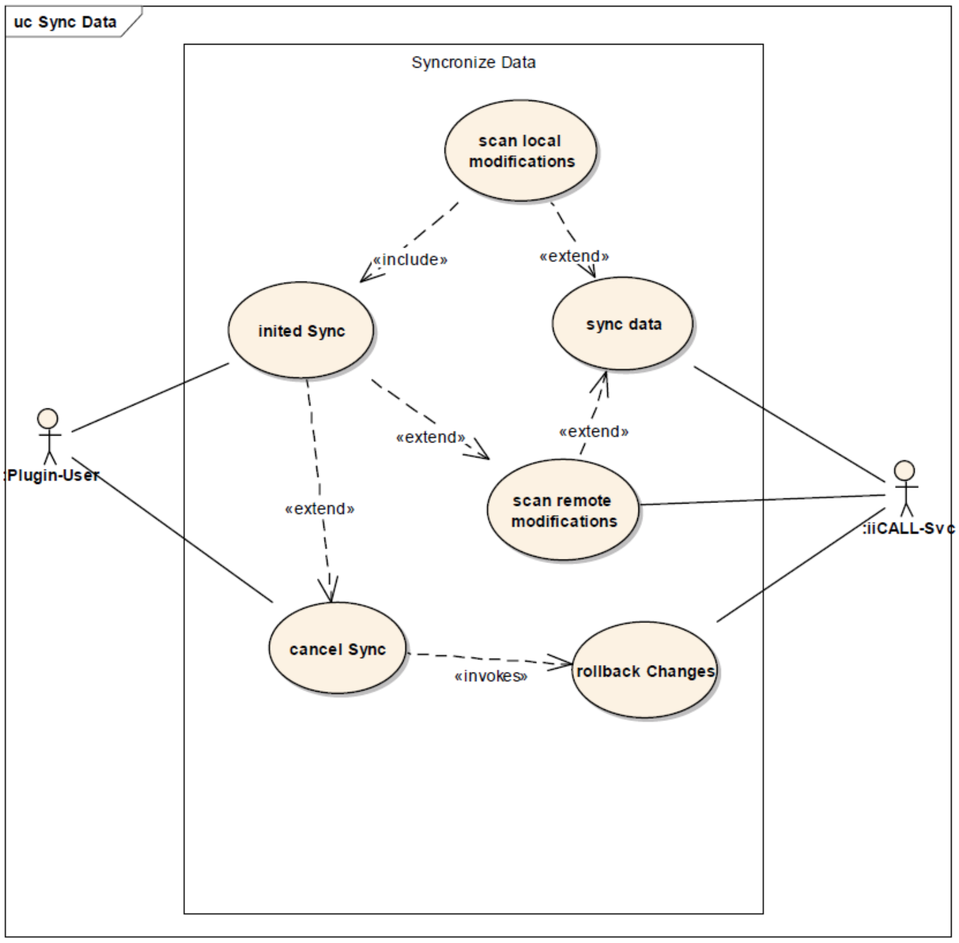


Figure 44. Use Case Diagram "Sync Data"

Use Case Initialize Sync. The user has started the data synchronization mechanism (see Table 36).

TABLE 36. Use Case Initialize Sync

Pre-condition	The user has started the data synchronization mechanism.
Basic path	Local changes are identified (see use case “Scan Local Modifications” in Table 37). The client connects to the iiCALL sync service at server side.
Alternate path	Remote changes are identified (see use case “Scan Remote Modifications” in Table 38). The client connects to the iiCALL sync service at server side.
Exceptions	The iiCALL sync service is not available. User is informed to try later. The synchronize process cannot be initialized.
Post-condition	The synchronize process is initialized. Use case “Sync Data” (see Table 39) follows.

Use Case Scan Local Modifications. Data synchronization has begun and changes to the local database need to be evaluated in order to only synchronize modified data (see Table 37).

TABLE 37. Use Case Scan Local Modifications

Pre-condition	The synchronize process is initialized (see use case “Initialize Sync” in Table 36).
Basic path	Changes of local database since last synchronization process are identified. The identified data is not empty. Use case “Sync Data” (see Table 39) follows.
Alternate path	There are no changes identified. User gets informed that client and server data are synchronized.
Exceptions	The iiCALL sync service is not available due to network timeout. User gets informed to try later. Process is stopped.
Post-condition	Data for synchronization are identified.

Use Case Scan Remote Modifications. All changes on the iiCALL server side will be evaluated and synchronize process (see use case "Sync Data" in Table 39) is started (see Table 38).

Table 38. Use Case Scan Remote Modifications

Pre-condition	The synchronize process is initialized (see use case "Initialize Sync" in Table 36).
Basic path	Changes of remote database since last synchronization process are identified. The identified data is not empty. Use case "Sync Data" (see Table 39) follows.
Alternate path	There are no changes identified.
Exceptions	The iiCALL sync service is not available due to network timeout. Process is stopped.
Post-condition	Data for synchronization are identified.

Use Case Sync Data. User wants to synchronize data. This process runs per default in dual mode, which means synchronizing data from server to client and from client to server. Care must be taken in order to not overwrite any pending changes, merging mechanism may be implemented (see Table 39).

Table 39. Use Case Sync Data

Pre-condition	The synchronize process is initialized (see use case "Initialize Sync" in Table 36).
Basic path	Use case "Scan Local Modifications" (see Table 37) identifies local changes. Remote data are updated
Alternate path	Use case "Scan Remote Modifications" (see Table 38) identifies remote changes. Local data are updated.
Exceptions	The iiCALL sync service is not available. The user gets informed to try later. Modifications cannot be synchronized.
Post-condition	All modifications are synchronized.

Use Case Cancel Sync. This mechanism cancels a current active data synchronization and performs a cleanup. Synchronization may be canceled by the user or because of a network timeout (see Table 40).

Table 40. Use Case Cancel Sync

Pre-condition	Synchronization is in progress
Basic path	Data transfer is stopped and already transferred data are identified. The identified data is not empty. Use case “Rollback Changes” (see Table 41) follows.
Alternate path	Already transferred data are empty.
Exceptions	The iiCALL sync service is not available due to network timeout. Cancellation request cannot be transmitted and already transferred data cannot be revoked.
Post-condition	Already transferred data for rollback are identified.

Use Case Rollback Changes. Perform rollback on uncommitted changes to the local and central database because of cancellation event (see use case “Cancel Sync” in Table 40) was triggered (see Table 41).

Table 41. Use Case Rollback Changes

Pre-condition	Use case “Cancel Sync” (see Table 40) is initialized.
Basic path	Transferred data is updated by former data.
Alternate path	not relevant
Exceptions	The iiCALL sync service is not available. The user gets informed to try later. Rollback cannot be fulfilled.
Post-condition	Rollback is fulfilled.

Use Case Diagram User Data. The use cases in the diagram “User Data” deal with configuration of user preferences (see Figure 45).

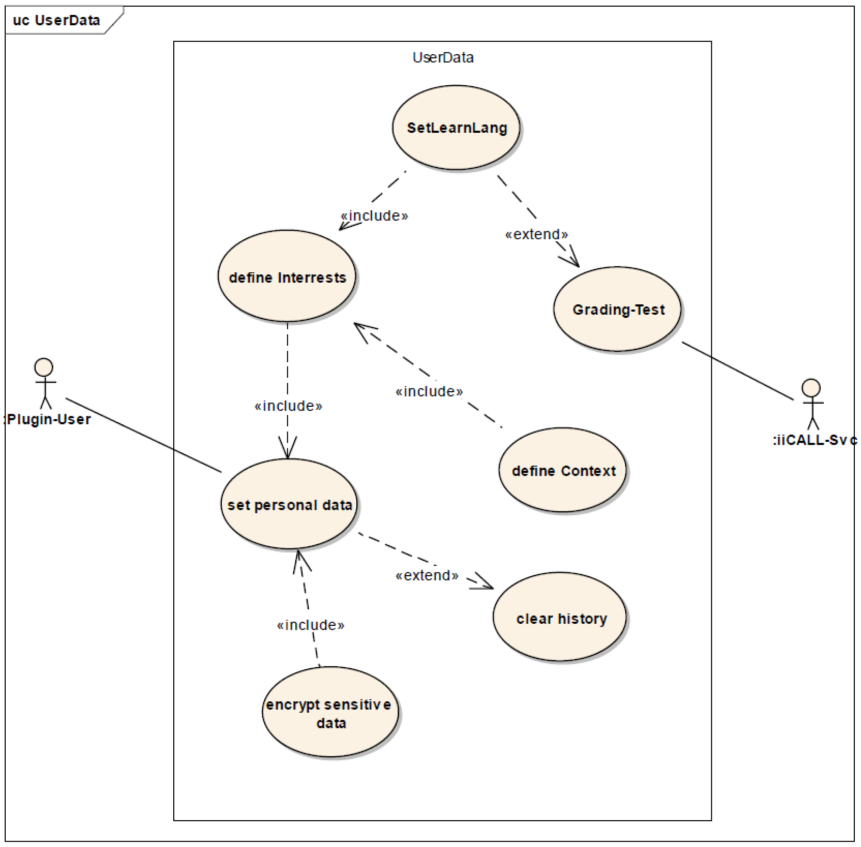


FIGURE 45. Use Case Diagram “User Data”

Use Case Set Personal Data. Personal data like name, email address, profile picture, and address shall be set by the user when using the iiCALL system (see Table 42).

TABLE 42. Use Case Set Personal Data

Pre-condition	The user wants to add or update his personal data.
Basic path	The user defines mandatory data name and email address, and optional data address and profile picture. The user has the option to share the information or keep it private.
Alternate path	not relevant
Exceptions	not relevant
Post-condition	Personal data are stored locally.

Use Case Define Interests. User defines interests within the iiCALL system. Interests consist basically of a language to learn and a context in which the language should be learned (see Table 43).

Table 43. Use Case Define Interests

Pre-condition	The user wants to add or update his interests.
Basic path	The user can define combinations of a language and an interest. It is possible to define different combinations for the same language and different combinations for the same interest. The list of available languages and interests is provided by the iiCALL system.
Alternate path	not relevant
Exceptions	not relevant
Post-condition	Interests are stored locally.

Use Case SetLearnLang. To efficiently use the system, the user has to specify a language he wants to learn. (see Table 44).

Table 44. Use Case SetLearnLang

Pre-condition	The user wants to add or update his learning language.
Basic path	User specifies preferred language from a list of available languages. Then the user has the possibility to define the language level. He can define it (see use case "Define Grading" in Table 12), or he can do a grading test (see use case "Do GradingTest" in Table 13).
Alternate path	not relevant
Exceptions	not relevant
Post-condition	Learning language is defined and stored locally.

Use Case Define Context. User has to define a context for which he wants to learn a foreign language. It is possible to define more than one context connected to the same or other languages (see Table 45).

TABLE 45. Use Case Define Context

Pre-condition	The user wants to add or update his preferred learning context.
Basic path	The user can define combinations of a language and a context. It is possible to define different combinations for the same language and different combinations for the same context. The context can be chosen from a list provided by the iiCALL system.
Alternate path	not relevant
Exceptions	not relevant
Post-condition	Preferred learning context is defined and stored locally.

Use Case Encrypt Sensitive Data. Encryption mechanism has to be implemented to encrypt sensitive information like account credentials to be safe from misuse by others (see Table 46).

TABLE 46. Use Case Encrypt Sensitive Data

Pre-condition	not relevant
Basic path	The client software has to ensure that sensitive data is stored only in encrypted form. Sensitive data are account credentials, passwords for workgroups (see use case "Set Group Password" in Table 50), etc.
Alternate path	not relevant
Exceptions	not relevant
Post-condition	Sensitive data is encrypted.

Use Case Clear History. Clear local or remote data to not leave any traces. For instance, this may be useful in case the user works on a public computer (see Table 47).

TABLE 47. Use Case Clear History

Pre-condition	The user wants to delete all information about his work.
Basic path	Local information is deleted, i.e. preferences stored in local database, test data, visited Web sites, and local cache.
Alternate path	Remote information is deleted, i.e. preferences stored in remote iiCALL database.
Exceptions	Relevant only for alternate path: The iiCALL service is not available. User gets informed to try later. Remote data cannot be deleted.
Post-condition	History data is deleted.

Use Case Diagram Workgroup. The use cases collected in the diagram “Workgroup” define functionalities around team learning issues. A user may be a member of several workgroups. Furthermore, a workgroup has to have at least one active member which may also be the moderator (see Figure 46).

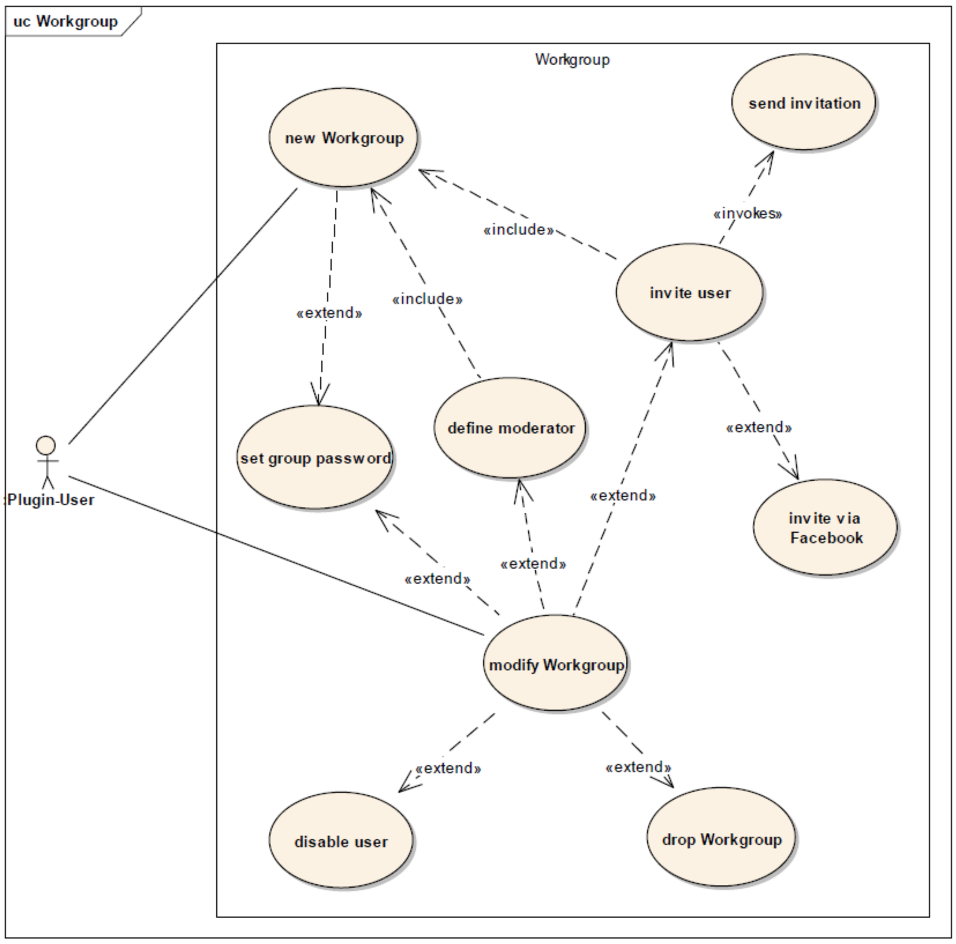


FIGURE 46. Use Case Diagram “Workgroup”

Use Case New Workgroup. A new workgroup shall be defined within the iiCALL system. The main goal of a workgroup is to connect people sharing the same interests (see Table 48).

TABLE 48. Use Case New Workgroup

Pre-condition	The user is registered in the iiCALL system.
Basic path	The user defines a name for the workgroup, which is checked for uniqueness by the iiCALL service. If the chosen name is not unique, the iiCALL service asks for a different name. Optionally, the user can add a workgroup description to specify the learning topic.
Alternate path	not relevant
Exceptions	The iiCALL workgroup service is not available. User gets informed to try later. Workgroup cannot be created.
Post-condition	Workgroup with a unique name and an optional workgroup description is created. The creator is the first moderator.

Use Case Modify Workgroup. A user can do some changes to a workgroup like inviting new users, setting a password, or defining a new moderator. Additionally, a member of the workgroup may be banned or the workgroup may be dropped as a whole (see Table 49).

TABLE 49. Use Case Modify Workgroup

Pre-condition	The user is moderator for a specific workgroup.
Basic path	User can invite others to join the workgroup. Use case “Invite User” (see Table 52) or use case “Invite via Facebook” (see Table 54) follow. The user can set or change a password for the workgroup. Use case “Set Group Password” (see Table 50) follows. User can define one or more moderators. Use case “Define Moderator” (see Table 51) follows. User can exclude members from the workgroup. Use case “Disable User” (see Table 55) follows. User can delete the workgroup. Use case “Drop Workgroup” (see Table 56) follows.
Alternate path	not relevant
Exceptions	The iiCALL workgroup service is not available. User gets informed to try later. Changes on workgroup cannot be fulfilled.
Post-condition	Workgroup configuration is changed.

Use Case Set Group Password. There should be the possibility to protect a workgroup by using a password, which means that only members knowing the password may join. It is optional to define a password or to keep a workgroup public (see Table 50).

TABLE 50. Use Case Set Group Password

Pre-condition	The user is moderator for a specific workgroup.
Basic path	The user defines a new password differing from the old one. An email is sent to all members and invited users of the workgroup with the new password. As there is no need for a comprehensive security, the password is sent as plain text in the email body.
Alternate path	The user wants to change the workgroup to be public and therefore, he deletes the actual password.
Exceptions	The iiCALL workgroup service is not available. User gets informed to try later. Password cannot be set.
Post-condition	New group password is set or password is deleted.

Use Case Define Moderator. For a workgroup a moderator has to be defined. By default, the creator of a workgroup is selected as moderator. This can be changed later. Only a moderator can grant moderator privileges to other members (see Table 51).

TABLE 51. Use Case Define Moderator

Pre-condition	A new workgroup has been created. The user is moderator for a specific workgroup. By default the creator of a workgroup is moderator.
Basic path	The user (moderator) chooses one or more members of the workgroup and grant moderator privileges to them.
Alternate path	not relevant
Exceptions	The iiCALL workgroup service is not available. User gets informed to try later. Moderator cannot be defined.
Post-condition	One or more moderators are defined.

Use Case Invite User. To join a workgroup, a user needs to have a valid invitation. An invitation can only be created by a moderator of a workgroup (see Table 52).

TABLE 52. Use Case Invite User

Pre-condition	The user is moderator for a specific workgroup.
Basic path	The user (moderator) invites another user to join the workgroup. He can invite other iiCALL users by defining username or email address. Use case "Send Invitation" in Table 53) follows.
Alternate path	The user (moderator) invites another user to join the workgroup. He can invite others by defining an email address. Use case "Send Invitation" in Table 53) follows.
Exceptions	The iiCALL workgroup service is not available. User gets informed to try later. Invitation is stopped.
Post-condition	User is invited or invitation is sent to an email address.

Use Case Send Invitation. Once an invitation has been created, the invited users will be notified. The invited user has the choice to accept or deny the invitation request (see Table 53).

TABLE 53. Use Case Send Invitation

Pre-condition	A moderator has started the use case "Invite User" (see Table 52).
Basic path	If the username is defined, the iiCALL invitation service will look for the corresponding email address of the invited user. An invitation email is sent to the given email address.
Alternate path	If email address is defined by the moderator, an invitation email is sent to the given email address. The invitation email includes information that one has to register for iiCALL before joining the workgroup is possible.
Exceptions	Username does not exist. The moderator gets informed that username does not exist. If email address does not exist for an already registered user, the moderator does not get informed.
Post-condition	Invitation email is sent.

Use Case Invite via Facebook. It is possible to invite people from different social network platforms like Facebook, Xing, etc. Therefore, an interface for each platform has to be created and should be flexible enough to include more networks when needed. Here, only Facebook invitation is described. (see Table 54).

Table 54. Use Case Invite via Facebook

Pre-condition	The user is moderator for a specific workgroup. The user must have a valid Facebook login.
Basic path	The user (moderator) invites other users via Facebook to join the workgroup. Thus, an invitation message is sent to one or more Facebook users, or even to a Facebook group. The invitation message includes information that one has to register for iiCALL before joining the workgroup is possible.
Alternate path	not relevant
Exceptions	The Facebook login mechanism is not working or offline. The user gets informed that the Facebook service is currently not available and to try it later. Invitation cannot be sent.
Post-condition	Invitation via a social network is sent.

Use Case Disable User. A user modifies a workgroup and wants to delete another user from this group. The affected user will be informed about being deleted from the workgroup (see Table 55).

Table 55. Use Case Disable User

Pre-condition	The user is moderator for a specific workgroup
Basic path	The user (moderator) chooses one or more users from the list of workgroup members and exclude them from the workgroup.
Alternate path	not relevant
Exceptions	The iiCALL workgroup service is not available. User gets informed to try later. Invitation is stopped.
Post-condition	User who should be deleted is excluded from the workgroup.

Use Case Drop Workgroup. A user wants to drop a workgroup. All members will be informed about the action (see Table 56).

TABLE 56. Use Case Drop Workgroup

Pre-condition	The user is moderator for a specific workgroup
Basic path	The user chooses one or more workgroups from the list of workgroups he is moderator for and deletes them. All members of all dropped workgroups get informed that the workgroup will not exist anymore. Additionally, all workgroup conversations will also be deleted.
Alternate path	not relevant
Exceptions	The iiCALL workgroup service is not available. User gets informed to try later. Workgroup cannot be dropped.
Post-condition	Workgroup is dropped.

9.2. Refined Message Information Model Representations

Following our idea of the iiCALL Development Framework, which is the iiCALL Development Process (see Figure 38 on page 70) and the iiCALL GDM (see Figure 37 on page 69), after verbal description of the use cases, we have to map the specified use cases to the Refined Message Information Model. All the classes are specified which support message communication and which, therefore, can be inherited from the iiCALL GDM. Here, we illustrate the mappings for the use cases “Workgroup”, “Grading Test”, “Login”, and “Chat”. For each of these use cases, we present a class diagram including cardinalities. The corresponding classes are defined showing their classCode and their attributes including data types [**Wahl and Winiwarter, 2014**].

Use Case “Workgroup”. A user can create a new workgroup and, by default, the creator becomes moderator for this workgroup. As moderator, the user can invite other users to join, he can modify the workgroup and set a password. For a complete specification of workgroup functionalities, see use case diagram “Workgroup” in Figure 46 on page 101. Figure 47 shows the corresponding R-MIM representation.

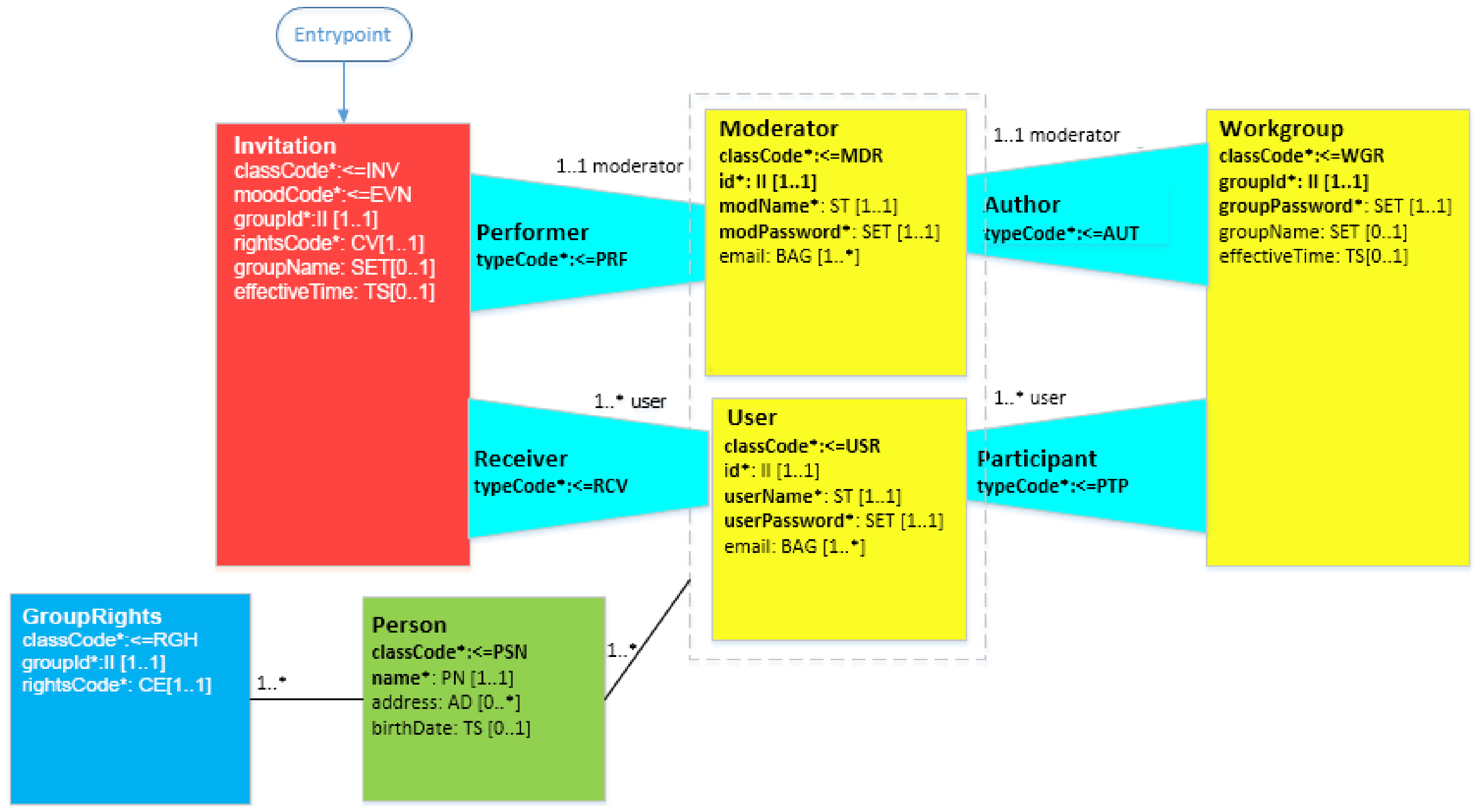


FIGURE 47. R-MIM Representation of Use Case “Workgroup”

The class "Person" stores personal data of a person, which are the name, the address and the date of birth (see Figure 48). It is inherited from the generic "Entity" class. Table 57 shows the attributes of class Person.

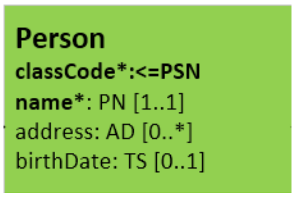


FIGURE 48. Class Person

TABLE 57. Attributes of Class Person

classCode	obligatory, 1..1, constant string: "PSN" encoded identifier of class
Name	obligatory, 1..1, (PN - personal name) full name of person fields of PN data type: • family name = family • given name = given • title = prefix qualifier • name addon = prefix qualifier
Address	optional, 0..*, (AD - address) full address of person fields of AD data type: • country code = country • zip code = postalCode • name of city = city • name of street = streetName • number = houseNumber
birthDate	optional, 0..1, (TS - timestamp) date of birth of person

The class “User” stores user data, which are an identification number, user credentials like username and password, and one or more email addresses (see Figure 49). It is inherited from the generic “Role” class. Table 58 shows attributes of the class User.

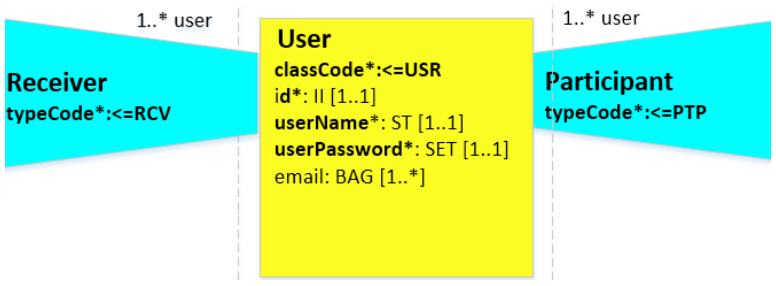


FIGURE 49. Class User

TABLE 58. Attributes of Class User

classCode	obligatory, 1..1, constant string: “USR” encoded identifier of class
id	obligatory, 1..1, (II - instance identifier) unique identifier of user
userName	obligatory, 1..1, (ST - character string)
userPassword	obligatory, 1..1, (SET - encrypted string)
email	obligatory, 1..1, (BAG - collection) the email addresses of user

The class “Moderator” stores information of a workgroup moderator, which are the identification number, user credentials like username and password, and one or more email addresses (see Figure 50). It is inherited from the generic “Role” class. Table 59 shows attributes of class Moderator.

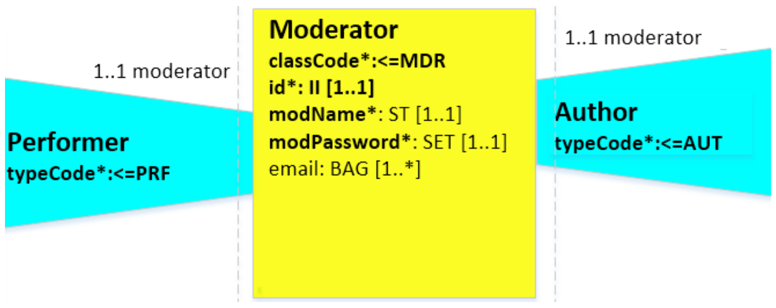


FIGURE 50. Class Moderator

Table 59. Attributes of Class Moderator

classCode	obligatory, 1..1, constant string: "MDR" encoded identifier of class
id	obligatory, 1..1, (II - instance identifier) unique identifier of moderator
modName	obligatory, 1..1, (ST - character string)
modPassword	obligatory, 1..1, (SET - encrypted string)
email	obligatory, 1..1, (BAG - collection) the email address of moderator

The class "GroupRights" is inherited from the generic class "State". It contains rights of a user in the corresponding working group (see Figure 51). Table 60 shows attributes of class GroupRights.

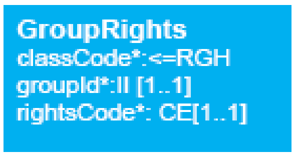


Figure 51. Class GroupRights

Table 60. Attributes of Class GroupRights

classCode	obligatory, 1..1, constant string: "RGH" encoded identifier of class
groupId	obligatory, 1..1, (II - instance identifier) unique identifier of workgroup
modName	obligatory, 1..1, (ST - character string)
rightsCode	obligatory, 1..1, (CV - Coded Value) User rights: • READ: Reading right • WRITE: Writing right • MOD: Moderator Encoded specification of workgroup user rights

The class "Workgroup" stores details of the workgroup, which are the workgroup's identification number, the password of the workgroup, and, if available, the workgroup's name and date of creation (see Figure 52). It is inherited from the generic "Role" class. Table 61 gives the attributes of class Workgroup.

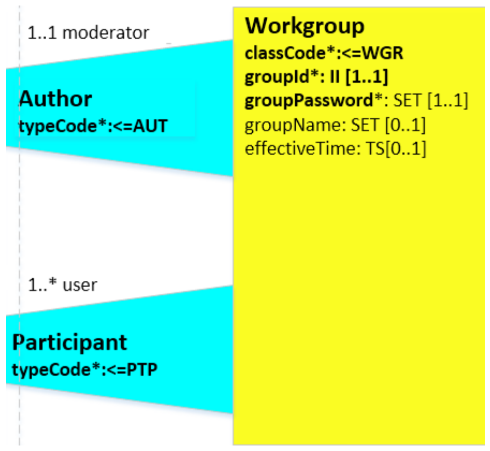


Figure 52. Class Workgroup

Table 61. Attributes of Class Workgroup

classCode	obligatory, 1..1, constant string: "WGR" encoded identifier of class
groupId	obligatory, 1..1, (II - instance identifier) unique identifier of workgroup
groupPassword	obligatory, 1..1, (ST - character string) password of workgroup
groupName	optional, 0..1, (SET - encrypted string) name of workgroup
effectiveTime	optional, 0..1, (TS - timestamp) creation time of workgroup

The class "Invitation" stores details of the invitation process of the workgroup's individual user or user group (see Figure 53). It is inherited from the generic "Activity" class. Table 62 shows attributes of class Invitation.

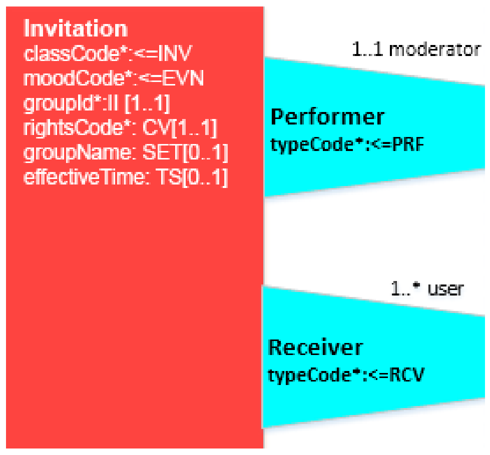


Figure 53. Class Invitation

Table 62. Attributes of Class Invitation

classCode	obligatory, 1..1, constant string: “INV” encoded identifier of class
moodCode	obligatory, 1..1, constant string: “EVN” - event
groupId	obligatory, 1..1, (II - instance identifier) unique identifier of workgroup
rightsCode	obligatory, 1..1, (CV - Coded Value) User rights: (READ: Reading right, WRITE: Writing right, MOD: Moderator) Encoded specification of workgroup user rights
groupName	optional, 0..1, (SET - encrypted string) name of workgroup
effectiveTime	optional, 0..1, (TS - timestamp) sending time of invitation

Use Case “Grading Test”. A grading test evaluates the language skill level of a user. In principle, grading tests are requested by the user. Alternatively, the user can assign his own skill level without passing a grading test. Adjustment of skill level must be possible in both directions, up and down. For a complete specification of grading test functionalities, see use case diagram “Grading Test” in Figure 40 on page 77.

The following skill levels should be predefined in the system:

- Level 1: Beginner
- Level 2: Basic
- Level 3: Advanced
- Level 4: Native Speaker
- Level 5: Teacher

In case the user wants to assign his personal skill level to 4 (Native Speaker) or 5 (Teacher), the system forces the user to pass an appropriate grading test. Figure 54 shows the corresponding R-MIM representation.

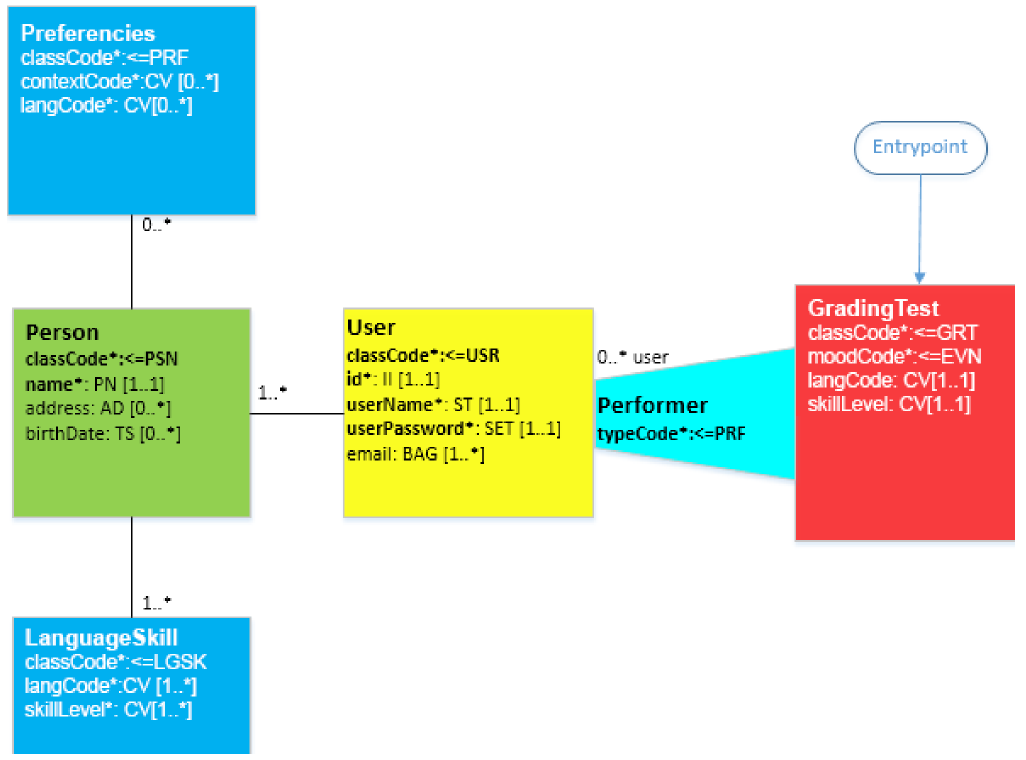


Figure 54. R-MIM Representation of Use Case "Grading Test"

The class “GradingTest” stores test information like the language of the grading test and the skill level the user has achieved (see Figure 55). It is inherited from the generic “Activity” class. Table 63 shows attributes of class GradingTest.

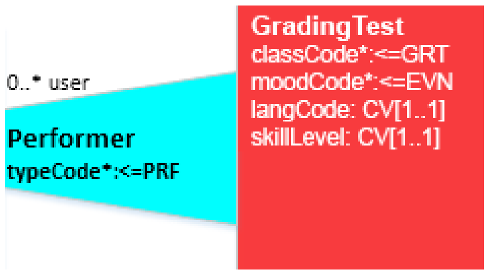


FIGURE 55. Class GradingTest

TABLE 63. Attributes of Class GradingTest

classCode	obligatory, 1..1, constant string: “GRT” encoded identifier of class
moodCode	obligatory, 1..1, constant string: “EVN” - event
langCode	obligatory, 1..1, (CV - Coded Value) Language code (e.g. ISO 639-1): • EN: English • DE: German • IT: Italian • FR: French encoded specification of language
skillLevel	obligatory, 1..1, (CV - Coded Value) Skill level: ((1)-(5)) encoded specification of skill level

The class “Preferences” stores interests of users, which are languages and contexts (see Figure 56). It is inherited from the generic “State” class. Table 64 gives attributes of class Preferences.

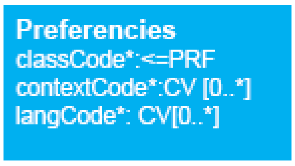


FIGURE 56. Class Preferences

TABLE 64. Attributes of Class Preferences

classCode	obligatory, 1..1, constant string: "PRF" encoded identifier of class
contextCode	optional, 0..*, (CV - Coded Value) Context: • MED: Medicine • SP: Sports • IT: Information Techology encoded learning context
langCode	optional, 0..*, (CV - Coded Value) Language code (e.g. ISO 639-1): (EN: English, DE: German, IT: Italian, FR: French) encoded specification of language

The class "LanguageSkill" holds the skill level the user has achieved by passing the grading test (see Figure 57). It is inherited from the generic "State" class. Table 65 shows attributes of class LanguageSkill.

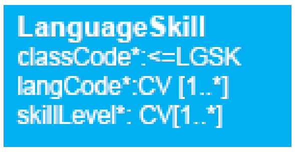


FIGURE 57. Class LanguageSkill

TABLE 65. Attributes of Class LanguageSkill

classCode	obligatory, 1..1, constant string: "LGSK" encoded identifier of class
langCode	obligatory, 1..*, (CV - Coded Value) Language code (e.g. ISO 639-1): (EN: English, DE: German, IT: Italian, FR: French) encoded specification of language
skillLevel	obligatory, 1..*, (CV - Coded Value) Skill level: ((1)-(5)) encoded specification of skill level

Use Case "Login". The usage of the iiCALL features is possible only for users who are logged in to the system. Therefore, each user has to register initially. Registration can be done using the internal iiCALL login mechanism or by linking a Facebook account. For a complete specification of grading test functionalities, see use case diagram "Grading Test" in Figure 39 on page 72. Figure 58 shows the corresponding R-MIM representation.

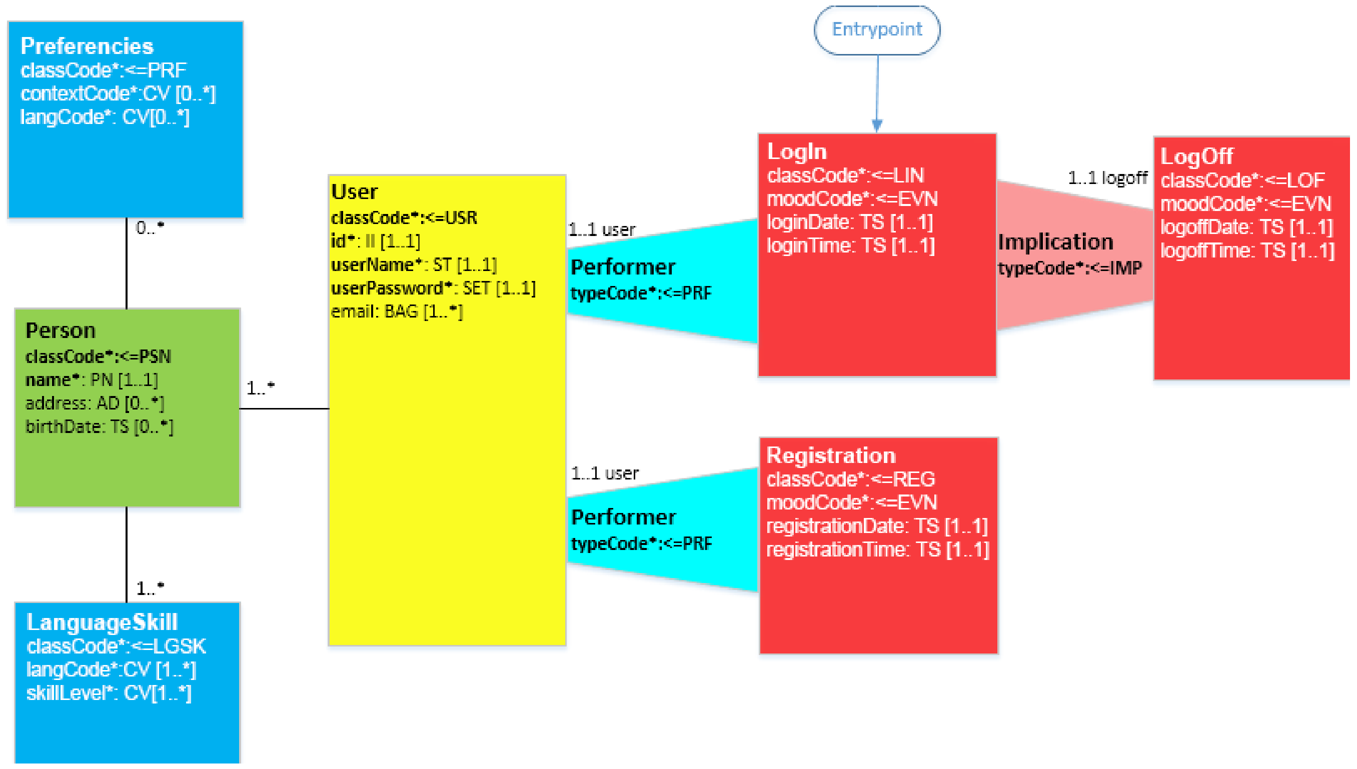


FIGURE 58. R-MIM Representation of Use Case “Login”

The class “LogIn” (see Figure 59) holds login information like date and time of login. Table 66 shows attributes of class LogIn.

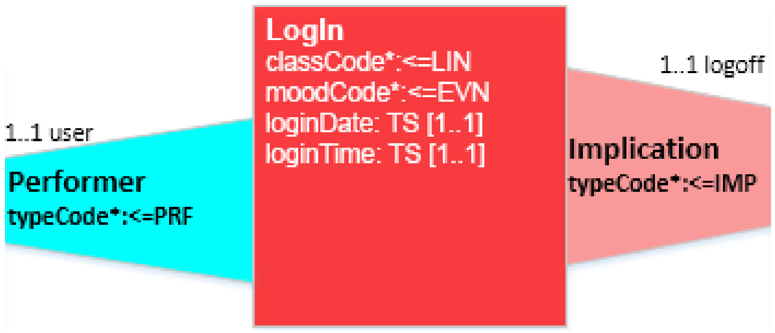


FIGURE 59. Class LogIn

TABLE 66. Attributes of Class LogIn

classCode	obligatory, 1..1, constant string: “LIN” encoded identifier of class
moodCode	obligatory, 1..1, constant string: “EVN” - event
loginDate	obligatory, 1..1, (TS - timestamp) date of login
loginTime	obligatory, 1..1, (TS - timestamp) timestamp of login

The class “LogOff” (see Figure 60) holds information of logout, i.e. date and time of logout. Table 67 shows attributes of class LogOff.

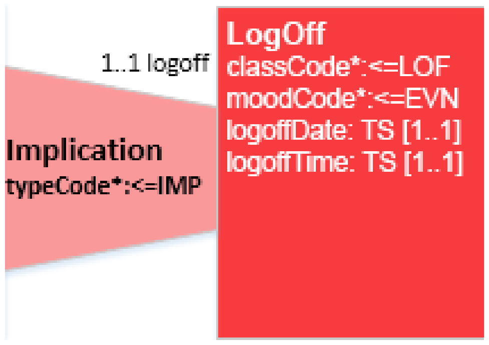


FIGURE 60. Class LogOff

Table 67. Attributes of Class LogOff

classCode	obligatory, 1..1, constant string: “LOF” encoded identifier of class
moodCode	obligatory, 1..1, constant string: “EVN” - event
loginDate	obligatory, 1..1, (TS - timestamp) date of logoff
loginTime	obligatory, 1..1, (TS - timestamp) timestamp of logoff

The class “Registration” (see Figure 61) holds information of registration, i.e. date and time of registration. Table 68 shows attributes of class Registration.

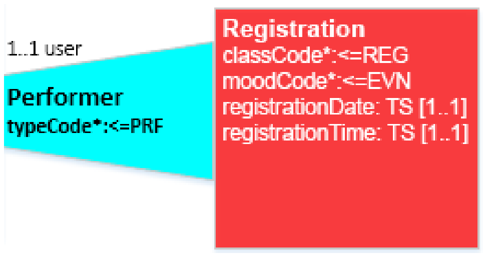


Figure 61. Class Registration

Table 68. Attributes of Class Registration

classCode	obligatory, 1..1, constant string: “REG” encoded identifier of class
moodCode	obligatory, 1..1, constant string: “EVN” - event
registrationDate	obligatory, 1..1, (TS - timestamp) date of registration
registrationTime	obligatory, 1..1, (TS - timestamp) timestamp of registration

Use Case “Chat”. Each user can start a chatting session. Chatting can happen privately or open for a workgroup. Therefore, the user invites others to join the chat. For a complete specification of chatting functionalities, see use case diagram “Chat” in Figure 42 on page 83. Figure 62 shows the corresponding R-MIM representation.

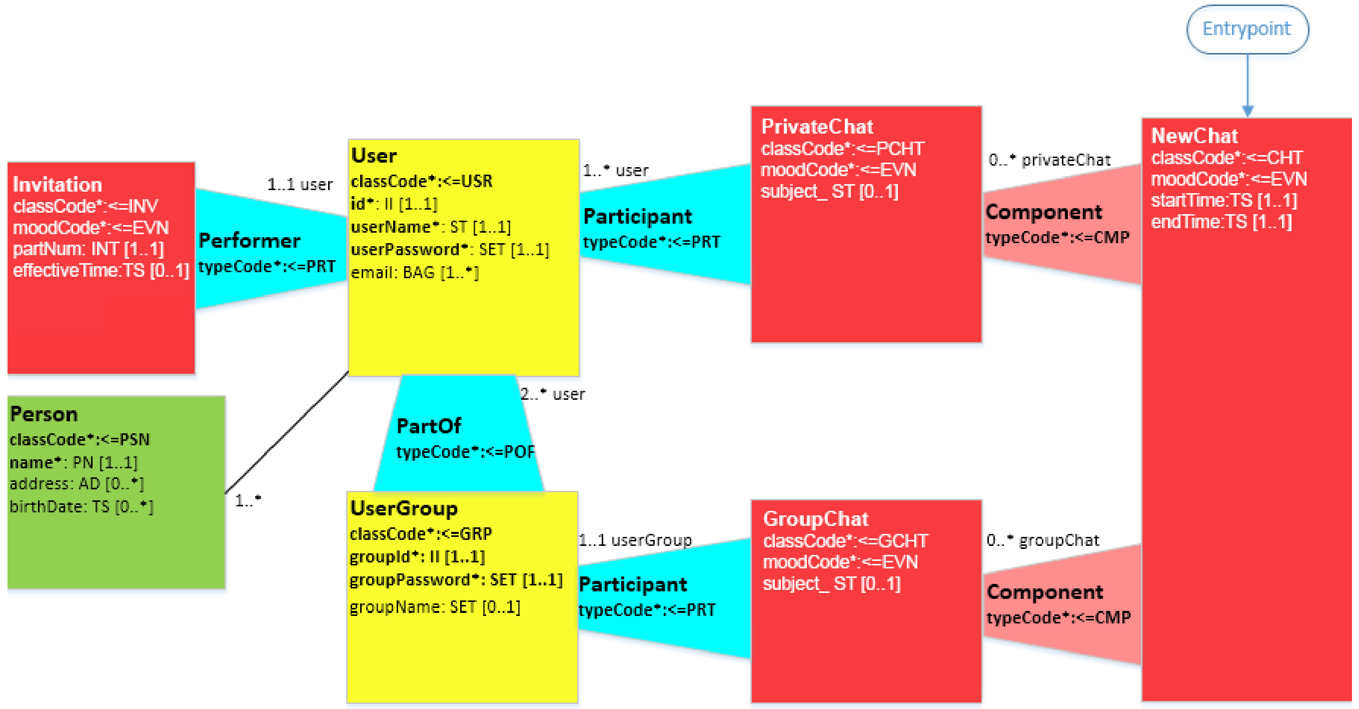


FIGURE 62. R-MIM Representation of Use Case "Chat"

The class “Invitation” is derived from the generic class “Activity” and contains detailed information on the invitation process of a private chat session (see Figure 63). For attributes of class Invitation see Table 69.

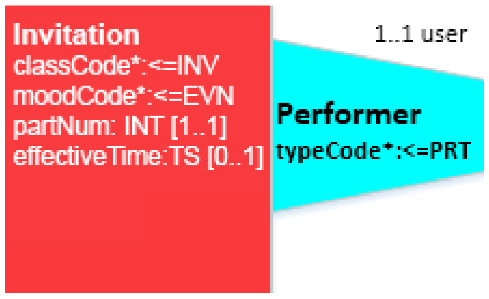


FIGURE 63. Class Invitation

TABLE 69. Attributes of Class Invitation

classCode	obligatory, 1..1, constant string: “INV” encoded identifier of class
moodCode	obligatory, 1..1, constant string: “EVN” - event
partNum	obligatory, 1..1, (INT - integer) number of chat members
effectiveTime	optional, 0..1, (TS - timestamp) sending time of invitation

The class “NewChat” is derived from the generic class “Activity” and stores starting time and end time of a chat (see Figure 64). For attributes of class NewChat see Table 70.

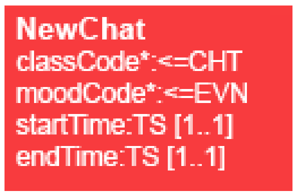


FIGURE 64. Class NewChat

TABLE 70. Attributes of Class NewChat

classCode	obligatory, 1..1, constant string: "CHT" encoded identifier of class
moodCode	obligatory, 1..1, constant string: "EVN" - event
startTime	obligatory, 1..1, (TS - timestamp) starting time of chat
endTime	obligatory, 1..1, (TS - timestamp) end time of chat

The class "GroupChat" is derived from the generic class "Activity" and stores information for a group chat session (see Figure 65). For attributes of class GroupChat see Table 71.

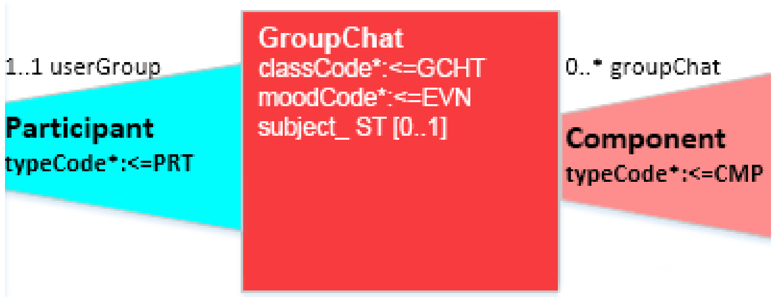


FIGURE 65. Class GroupChat

TABLE 71. Attributes of Class GroupChat

classCode	obligatory, 1..1, constant string: "GCHT" encoded identifier of class
moodCode	obligatory, 1..1, constant string: "EVN" - event
subject	optional, 0..1, (ST - character string) subject (heading) of chat

The class "PrivateChat" is derived from the generic class "Activity" and stores information for a private chat session (see Figure 66). For attributes of class PrivateChat see Table 72.

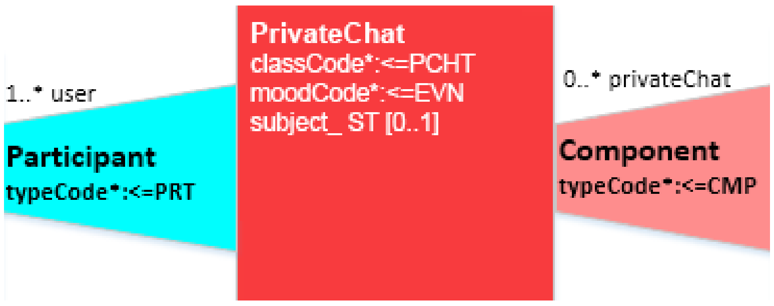


FIGURE 66. Class PrivateChat

TABLE 72. Attributes of Class PrivateChat

classCode	obligatory, 1..1, constant string: "PCHT" encoded identifier of class
moodCode	obligatory, 1..1, constant string: "EVN" - event
subject	optional, 0..1, (ST - character string) subject (heading) of chat

The class "UserGroup" is derived from the generic class "Role" and stores information of a user group that participates in a group chat session (see Figure 67). For attributes of class UserGroup see Table 73.

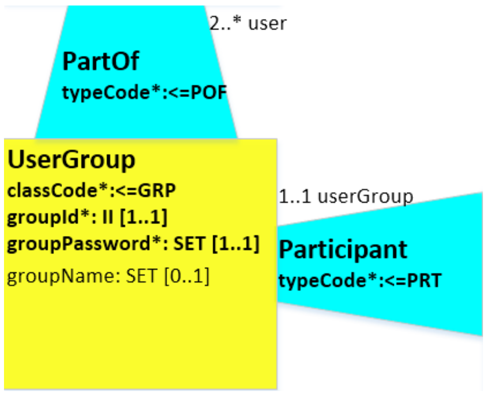


FIGURE 67. Class UserGroup

Table 73. Attributes of Class UserGroup

classCode	obligatory, 1..1, constant string: "GRP" encoded identifier of class
groupId	obligatory, 1..1, (II - instance identifier) unique ID of the group
groupPasswort	obligatory, 1..1, (SET - encrypted string) passwort of group
groupName	optional, 0..1, (SET - encrypted string) name of group

At this stage of requirements engineering, iiCALL core use cases are modeled through their UML representations. They are verbally described, and the mapping to the Reference Information Model defines the classes inherited from the core classes of the iiCALL Generic Data Model.

CHAPTER 10

The Automatic Exercise Learning Scenario

In [**Wahl et al., 2015b**], we firstly presented a typical use case for an iiCALL environment, the Automatic Exercise Learning Scenario. Here, we demonstrate the development of this learning scenario with respect to the phases of our iiCALL Development Process Version 1.

10.1. Phase 1: The Concept

Generating an automatic test and presenting it to the user is a typical learning scenario for the iiCALL environment. Figure 68 shows communication issues between involved systems using an UML sequence diagram. A sequence diagram can best address the proposed message exchange between involved systems [**Galler, 2015, Wahl et al., 2015b, Wahl et al., 2015a**].

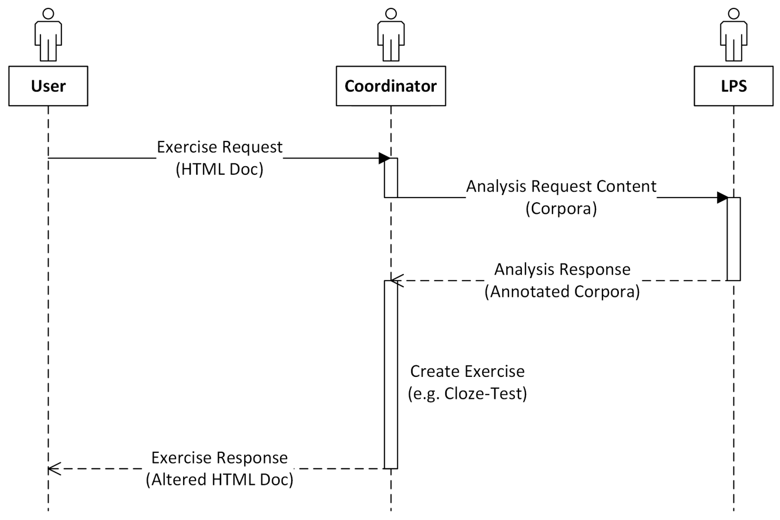

User
Coordinator
LPS
Exercise Request (HTML Doc)
Analysis Request Content (Corpora)
Analysis Response (Annotated Corpora)
Create Exercise (e.g. Cloze-Test)
Exercise Response (Altered HTML Doc)

Figure 68. UML Sequence Diagram for Automatic Exercise Learning Scenario

Starting on the left, the learner (i.e. the role of the user) owns a content on an arbitrary device that should act as basis for the language exercise the user requests. In case the content is available in textual form, a vocabulary test or a multiple-choice questionnaire could be imagined. These exercises are individually adapted and created. Thus, they are created based on the respective content by considering user preferences like language skills, interests, etc.

Requests for exercises are sent to a coordinator within the iiCALL environment. The coordinator is primarily responsible for the preparation of such exercises and manages user data, such as user preferences or learning progress. A request to the coordinator can contain the information that should serve as a basis for the language exercise. In the case of Figure 68, this is intended to be an HTML document, which the user has just opened in the Web browser. Alternatively to a direct transmission of the content, only a reference in form of a Uniform Resource Identifier (URI) can be transmitted. Here, the coordinator is responsible for retrieving the content. Additionally, the desired type of exercise must be told to the coordinator.

The coordinator will typically use one or more Language Processing Service (LPS)s to create such language exercises. Such services may be used to analyze or transform the content to be further processable. Usually, an LPS then uses varying NLP technologies. For example, content might be translated into another languages through such a service. It may also be responsible to assign the content to a specific context or classify certain parts with regard to the word's grammatical level. For this purpose, GATE will be contacted to fulfill NLP tasks. In addition, the use of lexical data like GermaNet, FrameNet, etc. is performed through the coordinator.

In the case of textual content, the communication between the coordinator and LPS will take the form of so-called linguistic corpora. In this context, such a corpus is to be understood as a structured representation of one or more texts, possibly given in one or more languages. To transmit back the result of an analysis to the coordinator, the LPS will annotate the corpus related to the calculated results. For instance, words or parts of text are marked with certain attributes and thus express a certain meaning.

When the coordinator has received all the necessary feedback messages from the LPS, the exercise can be created based on the previously obtained text and the user's preferences. This exercise will then be transmitted directly to the user in form of an HTML document. Alternatively, it is also possible to send the user a reference, i.e. a link (URI) to a prepared Web page holding the exercise. For instance, in case of multimedia data involved this will be the better option to reduce data volumes.

10.2. Phase 2: The Design

During the Design phase, the GDM-Core as described in Figure 37 on page 69 will now be adapted for the typical application. The overview of the refined model, i.e. the model with specialized classes derived from the GDM-Core, is shown in Figure 69 [**Galler, 2015, Wahl et al., 2015b, Wahl et al., 2015a**].

The classes Person, Endpoint, and Data are derived from the core class Entity. A person is identified by additional attributes indicating the name, the address, and the date of birth. Using the URI attribute of an Endpoint, different systems can be addressed (by reference) within an iiCALL environment. The class Data serves as a general container for data and contains the type, the way of encoding, and the data itself (represented as String).

The classes Document, User, Coordinator, and LPS are derived from the core class Role. A Document acts as a role for textual content. Documents belong to a specific language, they are encoded using a specific encoding type (ASCII, UTF-8, etc.), and they are represented by a fixed format (XML, HTML, etc.). User represents a user of an iiCALL environment and is identified by a user name and an email address. Storing a type of a user (teachers, administrators, etc.) and different permissions is possible, too. Class Coordinator and LPS have responsibilities as mentioned above. Additionally, they have the attribute capabilities, which represents a list of the capabilities and services that may be offered. For an LPS, this might be a list of possible analyses and transformations the LPS offers. In the case of the coordinator, it might be a list of possible exercises. Although LPS and Coordinator have the same attributes, they are mapped separately in the refined GDM since it is assumed that different attributes might be added for further developments.

The class Participation has no specific derived classes at the moment. It expresses the type of the relationship between a Role and an Action via the `typeCode`. Currently, participation is possible as a requester, receiver, provider, or subject.

Exercise and Analysis are derived from the class Act. In Exercise, attributes category and type characterize the type of exercise. This can be a cloze, a multiple-choice, etc. The presentation attribute defines the form in which the exercise is processed for the user. For example, a cloze test can be integrated into an existing text, whereas a vocabulary test may be presented to the user separately. The type of Analysis may be a translation, a classification, an annotation, etc. of text parts. Additionally, the `subType` can hold more specific classification information. Thus, in the case of annotation of a text, it can determine the word categories they refer to. The `moodCode` of the class Act defines the nature of an action, e.g. a question, a result, etc.

Finally, the ActRelation offers the possibility of concatenating several acts, therefore defining workflows. For example, one can use a corresponding `typeCode` to link the result of a query to its result.

10.3. Phase 3: The Implementation

During implementation phase, the defined requirements have to be implemented. Therefore, the iiCALL Software Framework was developed which will be explained in Chapter 11 via the iiCALL prototype. It offers on the one hand basic features for further implementations and on the other hand specific functionalities for the prototype.

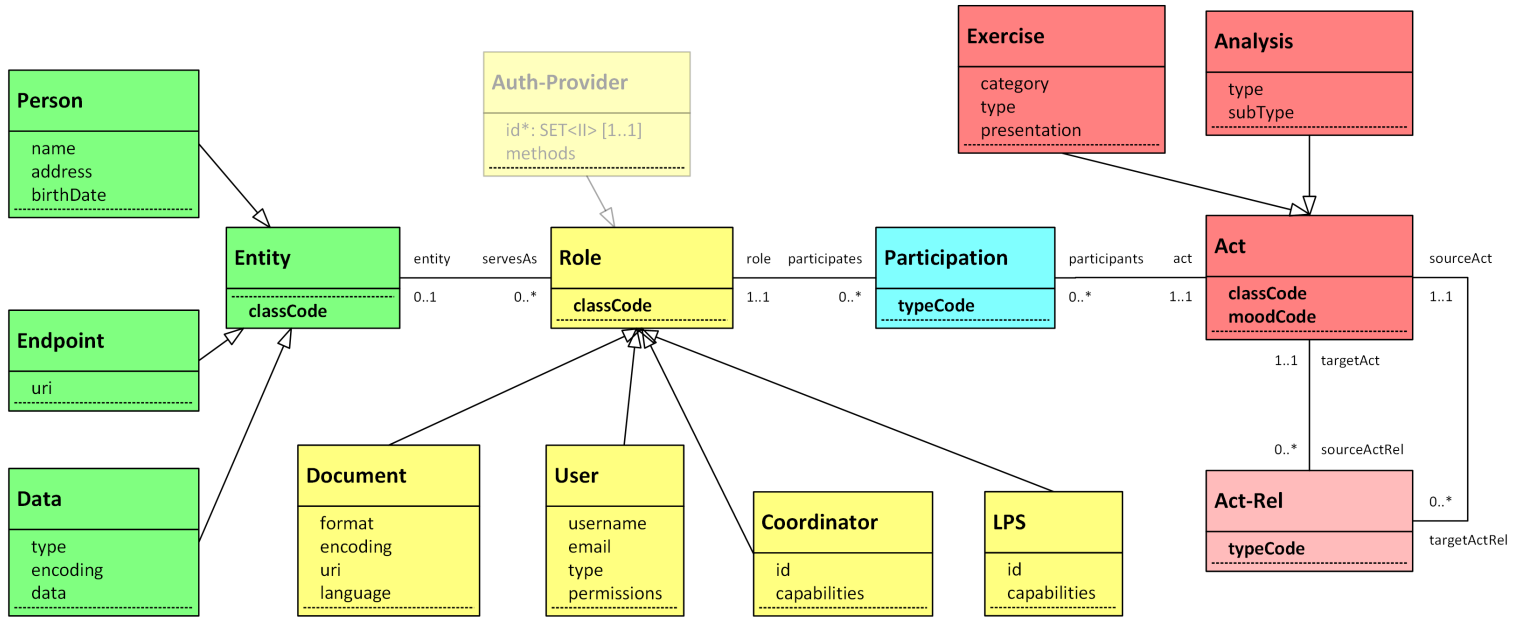


FIGURE 69. GDM Refined

CHAPTER 11

The iiCALL Prototype

Based on the concept and design of the Automatic Exercise Learning Scenario (see Chapter 10), here, we introduce the technical aspects of implementation [**Wahl et al., 2015b, Wahl et al., 2015a, Galler, 2015**].

11.1. The Prototype Architecture

Figure 70 shows the architecture of the iiCALL prototype. The learner (user) from the sequence diagram in Figure 68 uses a client which works with content from Web sites, emails or other documents. In order to meet the requirements of iiCALL to act as an integrated learning environment, the user might use different clients like Web browsers or email clients and, therefore, the connection to the learning environment must be integrated in the clients. For Web browsers such as Mozilla Firefox or Google Chrome, the development of corresponding plug-ins or add-ons is recommended. Similarly, this is also possible for some email clients like the Mozilla Thunderbird. The iiCALL prototype shows exemplary the development and functionalities within a plug-in for Mozilla Firefox.

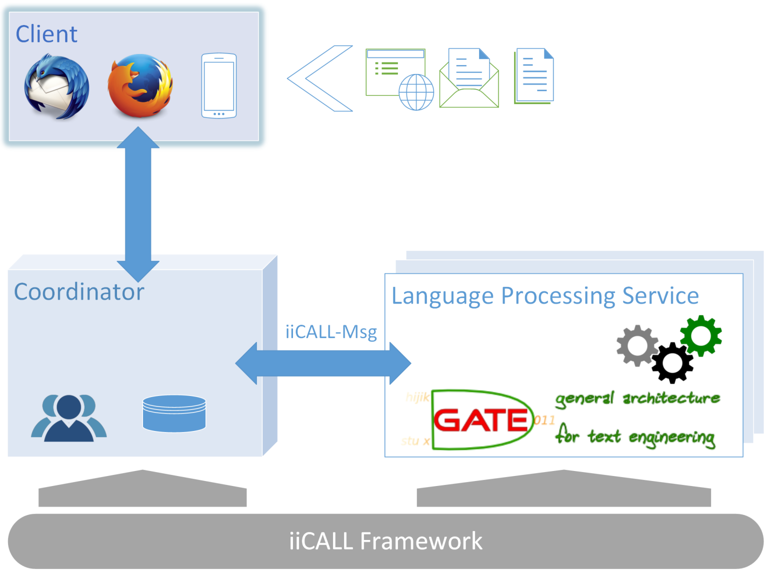


FIGURE 70. Prototype Architecture

The user visits a Web site in the browser and then he sends requests for the language exercises to the coordinator. The coordinator manages user data and accesses one or more LPSs to process the requests. For communication, the coordinator and the LPS use the underlying iiCALL framework.

This is a Java-based software framework, which essentially maps the GDM and provides further tools for exchanging iiCALL messages. The iiCALL prototype uses GATE for NLP functionalities, which are accessed through the LPS. Also the communication between the client and the coordinator uses iiCALL messages. As there is no client framework implementation available, here, iiCALL messages are generated by predefined message templates.

11.2. Used Technologies

The server part of iiCALL prototype is fully implemented in Java. Development is done using the Eclipse Java EE IDE for Web Developers with installed plug-ins Apache Ant[1] and Apache Ivy[2]. Apache Ant is a build tool where the building process is defined and configurable using an XML file. Apache Ivy handles dependencies of Java libraries. Additionally, Ivy is responsible for putting the iiCALL Software Framework into a local repository while building. Java Architecture for XML Binding (JAXB) is used for Java binding support. Therefore, Java classes are generated from an existing XML schema using the XML Java Compiler (XJC). The individual attributes of classes are established via corresponding Java annotations. This process is also called XMLBinding.

The client part of the prototype implements a Mozilla Firefox add-on. Therefore, the Mozilla AddOn SDK (also called Jetpack) is used to ease the usage of Web technologies like JavaScript, HTML, and Cascading Style Sheets (CSS) within the development process. For the usage of the SDK, Python is needed to be installed on the development system. The client part of the context-related vocabulary trainer was implemented using XML User Interface Language (XUL), which is also an easy way to create add-ons.

11.3. The iiCALL Software Framework

It is the primary task of the software framework to manage information exchange between various systems (like a Language Processing Service or coordinators) of an iiCALL environment. The software framework supports information exchange to ensure that the expected steps for transmitting and receiving iiCALL messages are transparent to the users, i.e. developers of applications. The software framework consists of different components with specific responsibilities and tasks. An overview is depicted by Figure 71. The software framework uses Java to implement the current prototype. The various elements will be discussed in more detail in the subsequent subsections [**Wahl et al., 2015a**].

Schema. The GDM stated in Figure 37 serves as a generic data model for the information interchange within the iiCALL environment. Furthermore, it establishes the overall foundation for the software framework. An XML schema definition (XSD) will be created around it. The representation comprises the basis and the policy for the auto-generation of Java classes, which represent the GDM. XSD is the favored technique to represent the

[1]`http://ant.apache.org/`
[2]`http://ant.apache.org/ivy/`

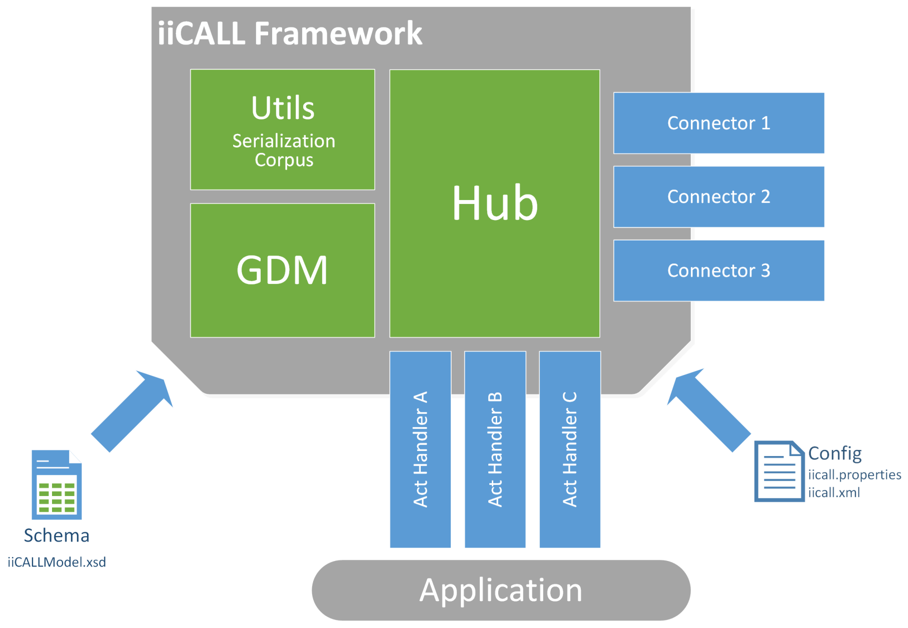


FIGURE 71. The iiCALL Framework Overview

GDM in this regard. It allows more flexibility compared to alternative methods such as the Document Type Definition (DTD). XSD is more convincing in this setting with benefits such as a strong type system and the compatibility to JAXB [**Wahl et al., 2015a**]. The complete schema definition can be viewed in Appendix A on page 173.

GDM. For the development of a framework for a universal iiCALL environment, mapping the GDM is an essential aspect. This procedure needs to be carried out automatically based on the XML schema. For every change in the GDM (which means adding new use cases) a corresponding implementation change in the framework will be required. The generation of Java classes from the GDM schema is automated by employing the JAXB. The automated procedure is depicted in detail in Figure 72 [**Wahl et al., 2015a**].

The build process of the framework integrates the mapping of the GDM. The Binding Compiler (XJC) derives the Java classes from the given GDM schema automatically before the source code is compiled. For the generated classes, each attribute is assigned to a different element of the schema. This complete process is referred as “Binding”. The framework operates with objects of these derived classes at the runtime, which in turn denote instances of elements of the GDM. Furthermore, JAXB-Context can be acquired by the several components of the framework as well as its users. This connection supports the transformation of instances into XML (marshal) and transformation from XML into instances (unmarshal). Because the GDM data needs to be first transformed into a suitable format to transmit it to different participating systems of the iiCALL environment, the transformation is an important feature. Additionally, the context extends the feature to verify data before the transformation process. This assures that the data

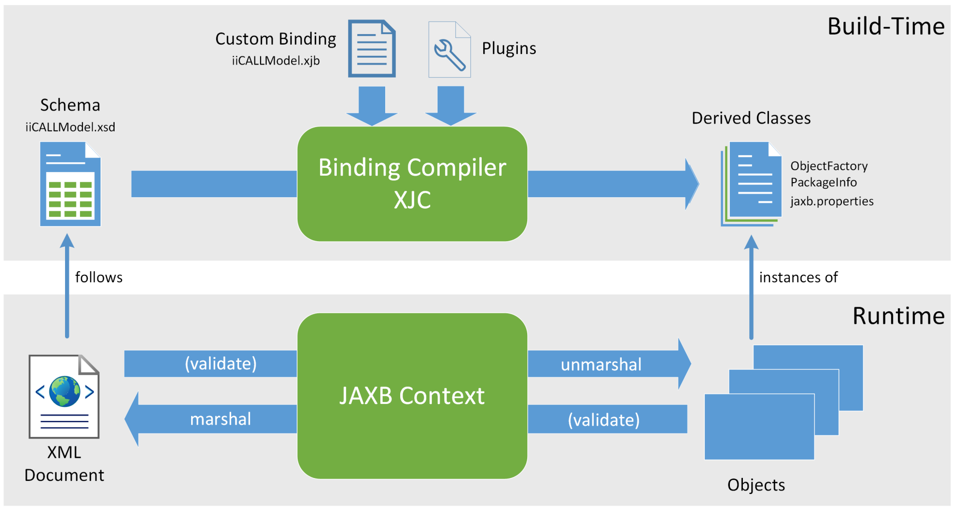


FIGURE 72. GDM Mapping

mimics the structure and policy defined in the schema. Thus, this guarantees the conformity with the underlying GDM [**Wahl et al., 2015a**].

Plug-ins. JAXB produces methods for all attributes to read and set its value (`get(), set()`) by default. For standard applications, this may be sufficient, but in this context, more methods are highly sought-after. Additional default methods to compare objects (`equal(), compare()`), to examine the data held by objects (`toString()`) or to duplicate objects (`clone()`) may be required by a developer who has already used the framework and its GDM classes. To add these methods manually after the generation process is not a reasonable option since this circumvents the aim to map the GDM in an automated manner. As a solution, [**Maraoui et al., 2012**] give various plug-ins that can be added to the binding compiler to create appropriate class methods automatically [**Wahl et al., 2015b**].

Custom Binding. Predefined directives to create classes from a given schema are employed by the JAXB compiler. This rule set is known as "Default Binding" and defines specifications such as the mapping of XML native types to Java types or the naming of classes and attributes. For the majority of cases, the default binding is adequate, but considering the framework in the prototype, it ought to be modified to solve name conflicts (see Listing 8). Available mechanisms related to similar Custom Bindings are discussed in [**Kawaguchi, 2008**] [**Wahl et al., 2015a**].

```
<jxb:bindings schemaLocation="iiCALLModel.xsd">
 <jxb:bindings node="//xs:element[@name='iicall-msg']">
  <jxb:class name="IICALLMsg" />
 </jxb:bindings>
</jxb:bindings>

<jxb:globalBindings>
 <xjc:superClass name="fhtw.iicall.framework.ModelRoot" />
```

```
</jxb:globalBindings>
```

LISTING 8. Part of the iiCALL Custom Binding (iiCALLModel.xjb) [**Galler, 2015**]

Connector. Connectors are flexible connections between the participating systems in an iiCALL environment. The primary job of the connector is to receive messages and forward them to the hub of the framework. Inversely, messages will be accepted from the hub of the framework and transmitted to opposite connectors of different systems. For this reason, a connector chooses the parameter for communication, particularly for transmission protocol and message encoding format. Subsequently, connectors organize communication in a transparent way to all the users of the framework [**Wahl et al., 2015a**].

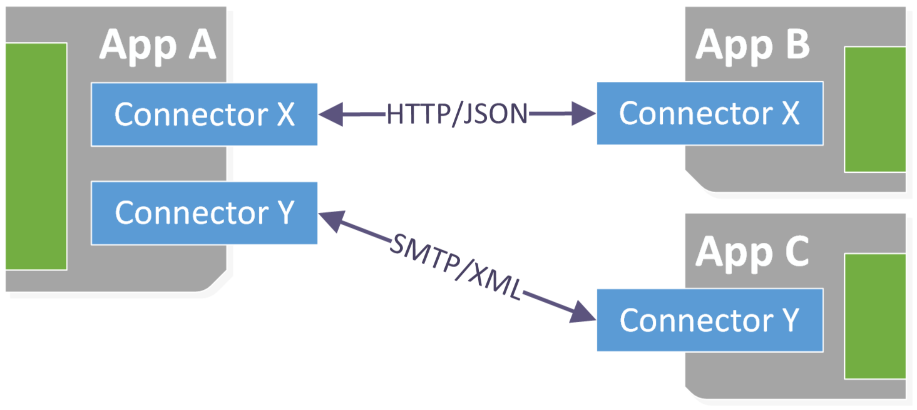


FIGURE 73. Message Exchange via Connectors

Figure 73 demonstrates this method. App A and App B can communicate via Connector X. The connector uses HTTP as the transport protocol and JSON format for encoding the GDM data. Subsequently, App A communicates with App C by utilizing Connector Y, which transmits messages in XML format using the SMTP transport protocol. Usually, connectors are delivered as components within the framework. Additionally, it is possible for the users of the framework to utilize their own specific connectors if needed [**Wahl et al., 2015a**].

Act-handler. Act-handlers allows the users of the framework to execute expected behaviors and duties associated with a certain activity. To achieve this, the framework will pass an activity (encapsulated in the received message) to the act-handler. The built-in logic inside the act-handler will investigate the complete activity and initialize further actions. The act-handler will create a new activity and pass it to the framework if the received message activity needs a response. Alternatively, the present activity can be altered and returned. The coordinator shown in the UML sequence diagram in Figure 68 on page 124, for instance, may use an act-handler to execute logic to handle requests for exercises and to generate the exercise. An LPS may use an act-handler to serve requests for NLP-related analysis [**Wahl et al., 2015a**].

Hub. Between connectors and act-handlers, the hub acts as an intermediary for acts and messages. The primary job of the connector is to receive a message, de-serialize it, and then forward it to the hub.

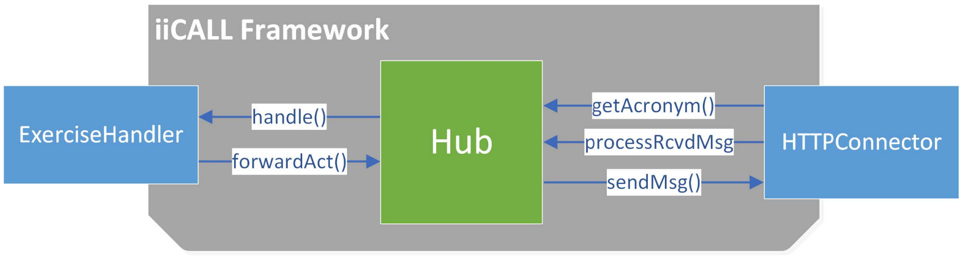


FIGURE 74. Act-handler, Hub and Connector Interaction

The hub will examine the message for encapsulated activities within the received message. Depending on the activity and the data contained within, the hub can determine the act-handler that executes the activity. Inversely, an act-handler may forward a prepared activity to the hub. The hub examines the data and structure of this activity with the encapsulated contents to determine the course of processing. So, if the activity contains an endpoint entity (refer to Figure 69 on page 127) which designates a certain system as receiver for that present activity, the hub will decide the connector to which it will forward the activity for transmission. The connector, in turn, knows how to address the system by the information given within the endpoint element (see Figure 74) [**Wahl et al., 2015a**].

Compared to the sequence from Figure 68 on page 124, the ExerciseHandler of the coordinator receives a request for an exercise in the form of an act from the hub (by method `handle()`). The LPS processes this request. Therefore, the ExerciseHandler creates a new Act of the type Analysis (see Figure 75). This Act now formulates a query (`moodCode = RQO`) for an analysis (`classCode = ANLZ`). By means of a corresponding participation (provider, `typeCode = PRV`), an LPS is assigned to the action as the provider of such service. This LPS is addressed via a concrete endpoint (`classCode = ENP`). This endpoint is known to the coordinator and thus, also to the exercise handler.

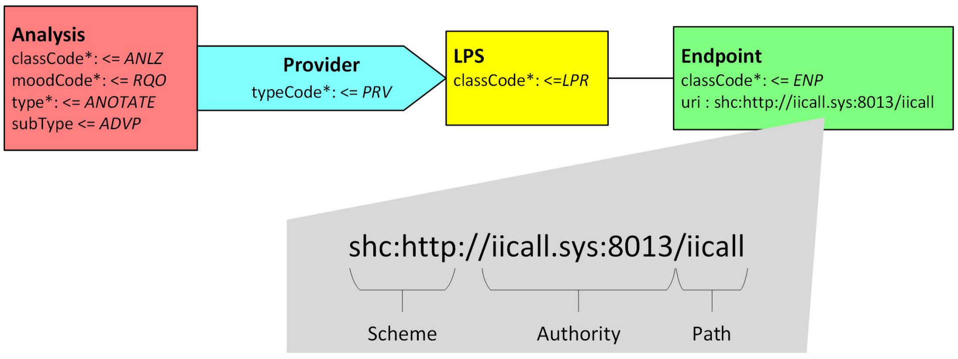


FIGURE 75. Analysis to Endpoint Addressing

Configuration. To adapt the framework to the requirements of its users, two layers of configuration are possible. The first layer of configuration is obligatory and is contained in an XML file. It defines the connectors and act-handlers needed for a certain framework instance. Furthermore, the act-handlers are allocated to analogous activities. The second layer of configuration is optional and includes parameters for the numerous components of the framework. For instance, a second layer configuration may set the listening port to be used by a certain connector. In case the second layer configuration is not given by the user, the framework will utilize default values [**Wahl et al., 2015a**]. Configuration files are shown in Appendix B on page181.

Utilities. Utility classes support framework developers as well as framework users by rendering practical methods for general routines to increase efficiency. Figure 71 illustrates two classes of such kind: Serialization and Corpus. The first class (Serialization) implements several methods concerning the transformation of iiCALL messages from and into the JSON or XML-format, respectively. These methods are especially valuable for the implementation of connectors. The second class (Corpus) implements several methods to transform annotated corpora into formats that are suitable for transmission. These methods allow the corpora to be associated with iiCALL messages [**Wahl et al., 2015a**].

For instance, the class Corpus provides various methods. The method `toGateXML()` converts the corpus to a GATE-specific, XML-based representation. The `toAnnotatedDoc()` routine supplies the underlying text, with the annotation taking place directly in the text using tags. Method `toXCESAnno()` returns the annotations to the texts of a corpus according to the Corpus Encoding Standard for XML (XCES) standard[3]. In addition, `toPlainText()` returns the raw text in the unedited form to which the XCES annotations refer. The methods `loadGateXML()` and `loadXCES()` work in the opposite direction by accepting a corpus of the respective format as a string and returning a Java object of type Document. The following text, which can be returned by executing the method `toPlainText()`, gives a sample text to illustrate these methods.

> "Far far away, behind the word mountains, far from the countries Vokalia and Consonantia, there live the blind texts. Separated they live in Bookmarksgrove right at the coast of the Semantics, a large language ocean."[4]

The method `toAnnotatedDoc()` returns the text annotated as shown in Listing 9.

```
<paragraph>
<ADVP>Far far away</ADVP>, <PP>behind</PP> the word
    mountains, <ADVP>far</ADVP> <PP>from</PP> the countries
    Vokalia and Consonantia, <ADVP>there</ADVP> live the
    blind texts. Separated they live <PP>in</PP>
    Bookmarksgrove right <PP>at</PP> the coast <PP>of</PP>
    the Semantics, a large language ocean.
```

[3]`http://www.xces.org/`

[4]taken from `http://webdesignernotebook.com/examples/css-text.html`

```
</paragraph>
```

LISTING 9. Annotated Sample Text: Method toAnnotatedDoc() [**Galler, 2015**]

The method `toXCESAnno()` returns a specific XCES represention, which is shown in Listing 10.

```
<?xml version='1.0' encoding='UTF-8'?>
<cesAna xmlns="http://www.xces.org/schema/2003" version="1.0
    ">
 <struct type="ADVP" from="0" to="12" n="299"/>
 <struct type="PP" from="14" to="20" n="300"/>
 <struct type="ADVP" from="41" to="44" n="302"/>
 <struct type="PP" from="45" to="49" n="303"/>
 <struct type="ADVP" from="89" to="94" n="305"/>
 <struct type="PP" from="137" to="139" n="311"/>
 <struct type="PP" from="161" to="163" n="313"/>
 <struct type="PP" from="174" to="176" n="315"/>
 <struct type="PP" from="248" to="250" n="321"/>
</cesAna>
```

LISTING 10. Sample Text in XCES: Method toXCESAnno() [**Galler, 2015**]

Figure 76 illustrates an example of how a corpus is inserted into an iiCALL message using above mentioned methods. The procedure follows in principle the idea of the Multipurpose Internet Mail Extensions (MIME) as specified in RFC2045 [**Freed and Borenstein, 1996a**] and RFC2046 [**Freed and Borenstein, 1996b**]. The advantage of this approach is the possibility of integrating multimedia content in iiCALL messages, in addition to linguistic corpora [**Galler, 2015**].

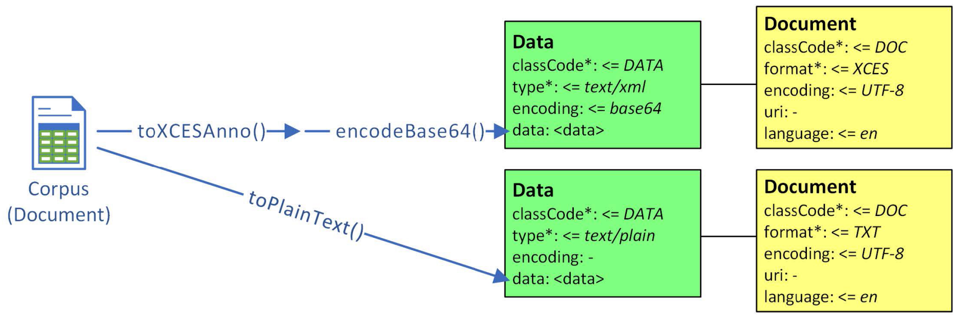


FIGURE 76. A Corpus inside an iiCALL Message

Inclusion. The inclusion of the framework requires the following three steps for its users:

- The framework is included as a library, so it needs to be added to the classpath of the user's application.
- The second step incorporates the implementation of act-handlers for particular activities and the configuration of the framework.
- The framework requires being instantiated and initialized at runtime by calling corresponding methods.

Instantiation and initialization of the framework at runtime is exemplary illustrated in Listing 11. For this purpose, primarily, an instance of the class `IICALL` is created (see line 4). It acts as the interface to the framework. The framework is started by calling the method `start()`. Here, the configuration files are read, and all act-handlers and connectors of the configuration files are initialized. By the method `getConfig()` and the method `getProperties()`, an application can access the configuration of the iiCALL framework at runtime.

```
 1 private static IICALL iiCallInstance;
 2
 3 public static void main(String[] args) throws Exception {
 4  iiCallInstance = new IICALL();
 5  iiCallInstance.start();
 6
 7  Runtime.getRuntime().addShutdownHook(new Thread() {
 8   public void run() {
 9    iiCallInstance.stop();
10   }
11  });
12
13  iiCallInstance.getConfig();
14  iiCallInstance.getProperties();
15
16  //Further Application Code
17 }
```

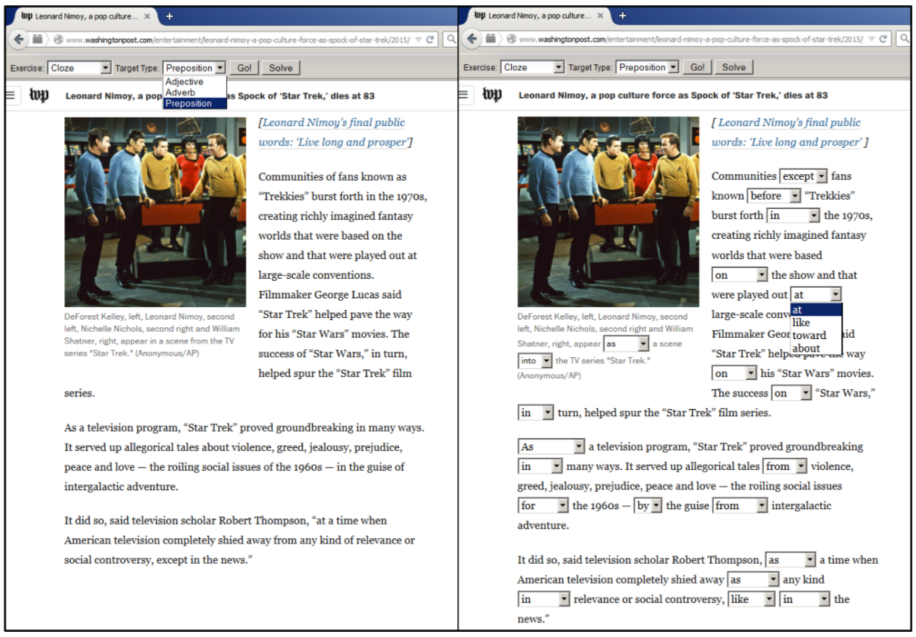
Listing 11. Inclusion of the Framework at Runtime [**Galler, 2015**]

11.4. The Prototype Demonstration

The automatic exercise learning scenario as described in Chapter 10 was implemented as a working prototype. The user perspective describes the handling of the prototype, and the technical perspective exchanges of iiCALL messages [**Galler, 2015, Wahl et al., 2015a**].

The User Perspective. From the user's point of view, the user starts by visiting a Web site containing various articles. On the left side of Figure 77 [**Wahl et al., 2015a**], a typical browser content is shown. Firstly, the user has to install the iiCALL browser extension. The GUI of the extension is shown as additional content bar at the top of the browser. On visiting a specific article that should act as textual content for the learning scenario, the user can start the exercise. In the left-hand drop-down list the user can select the desired type of exercise. In the second drop-down list, the user defines the type of word for which the language exercise will be created. In the specific case, the user selects a cloze test for prepositions in the current article.

By pressing the "Go!" button, the browser extension sends a corresponding request for a language exercise to the iiCALL coordinator. Therefore, it sends a constructed iiCALL message. The coordinator forwards the request together with the present text to the LPS. The LPS processes data and


Leonard Nimoy, a pop culture...
www.washingtonpost.com/entertainment/leonard-nimoy-a-pop-culture-force-as-spock-of-star-trek/2015/
Exercise: Cloze
Target Type: Preposition
Adjective
Adverb
Preposition
Go!
Solve

Leonard Nimoy, a pop culture force as Spock of 'Star Trek,' dies at 83

[Leonard Nimoy's final public words: 'Live long and prosper']

DeForest Kelley, left, Leonard Nimoy, second left, Nichelle Nichols, second right and William Shatner, right, appear in a scene from the TV series "Star Trek." (Anonymous/AP)

Communities of fans known as "Trekkies" burst forth in the 1970s, creating richly imagined fantasy worlds that were based on the show and that were played out at large-scale conventions. Filmmaker George Lucas said "Star Trek" helped pave the way for his "Star Wars" movies. The success of "Star Wars," in turn, helped spur the "Star Trek" film series.

As a television program, "Star Trek" proved groundbreaking in many ways. It served up allegorical tales about violence, greed, jealousy, prejudice, peace and love — the roiling social issues of the 1960s — in the guise of intergalactic adventure.

It did so, said television scholar Robert Thompson, "at a time when American television completely shied away from any kind of relevance or social controversy, except in the news."

Leonard Nimoy, a pop culture force as Spock of 'Star Trek,' dies at 83

[Leonard Nimoy's final public words: 'Live long and prosper']

DeForest Kelley, left, Leonard Nimoy, second left, Nichelle Nichols, second right and William Shatner, right, appear as a scene into the TV series "Star Trek." (Anonymous/AP)

Communities except fans known before "Trekkies" burst forth in the 1970s, creating richly imagined fantasy worlds that were based on the show and that were played out at large-scale conv at like toward about id "Star Trek" help way on his "Star Wars" movies. The success on "Star Wars," in turn, helped spur the "Star Trek" film series.

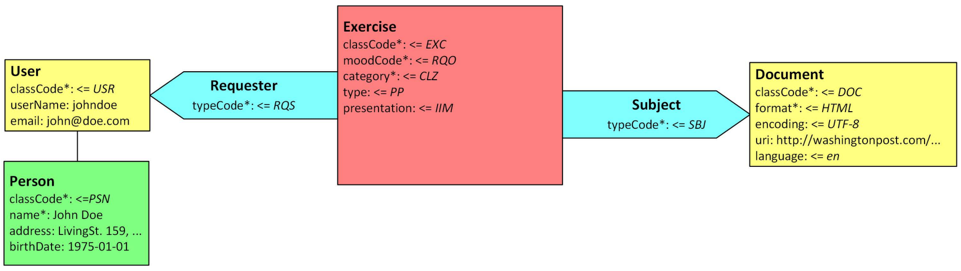
As a television program, "Star Trek" proved groundbreaking in many ways. It served up allegorical tales from violence, greed, jealousy, prejudice, peace and love — the roiling social issues for the 1960s — by the guise from intergalactic adventure.

It did so, said television scholar Robert Thompson, as a time when American television completely shied away as any kind in relevance or social controversy, like in the news."

FIGURE 77. Prototype Demonstration

returns a corpus annotated with all contained prepositions. The coordinator can create an exercise with this corpus and sends it back to the user. The browser extension integrates the exercise into the original article as a multiple choice cloze test (see the right side of Figure 77). The user can now try to select the correct prepositions from the lists containing a correct and several incorrect answers. Finally, pressing the "Solve" button, the extension returns the percentage of correct answers.

The Technical Perspective - iiCALL Messages. This section explains the technical view of the prototype with particular focus on the messages which are exchanged between the individual systems of the prototype. Figure 78 shows the message which will be sent from the Web browser plug-in to the coordinator.


User
classCode*: <= *USR*
userName: johndoe
email: john@doe.com

Person
classCode*: <=*PSN*
name*: John Doe
address: LivingSt. 159, ...
birthDate: 1975-01-01

Requester
typeCode*: <= *RQS*

Exercise
classCode*: <= *EXC*
moodCode*: <= *RQO*
category*: <= *CLZ*
type: <= *PP*
presentation: <= *IIM*

Subject
typeCode*: <= *SBJ*

Document
classCode*: <= *DOC*
format*: <= *HTML*
encoding: <= *UTF-8*
uri: http://washingtonpost.com/...
language: <= *en*

FIGURE 78. iiCALL Message from the Client to the Coordinator

The core of this message is an exercise-act, which formulates a query (`moodCode = RQO`) for a cloze test (`category = CLZ`). The attribute `type` specifies the type of words to be used for the cloze test. In the example, the exercise for prepositions is selected (`type = PP`). Using the attribute `presentation`, the client specifies the form in which the cloze test is expected. The value `IIM` (short form for Integrated Instant Markup) defines that the exercise is to be created as an HTML document, which can be directly integrated into the Web site of the client. The browser extension assigns the user the role of a requester (which is a Participation, identified by `typeCode = RQS`) and adds user data (Role: User) as well as data about the person (Entity: Person) to the request. The subject for the cloze test (Participation, `typeCode = SBJ`) and the currently viewed article can be transferred either as a Data-Entity or as a link to a URI in a Document-Role. In the shown example, the address of the article is set in the attribute `uri`. The format for the document is marked with the attribute `format` with value `HTML`. The language and encoding are taken from the metadata of the document head and are also specified in the attributes of the Document-Role (`language, encoding`).

The message from Figure 78 is sent to the coordinator. Since the content for the exercise is referenced by an URI, the Web page must be retrieved using an HTTP request. It is the task of the coordinator to filter the resulting Web page for relevant texts.

The handler of the coordinator now generates a request for a linguistic analysis, which is subsequently sent to the LPS based on the filtered text. The individual elements of this message are shown in Figure 79.

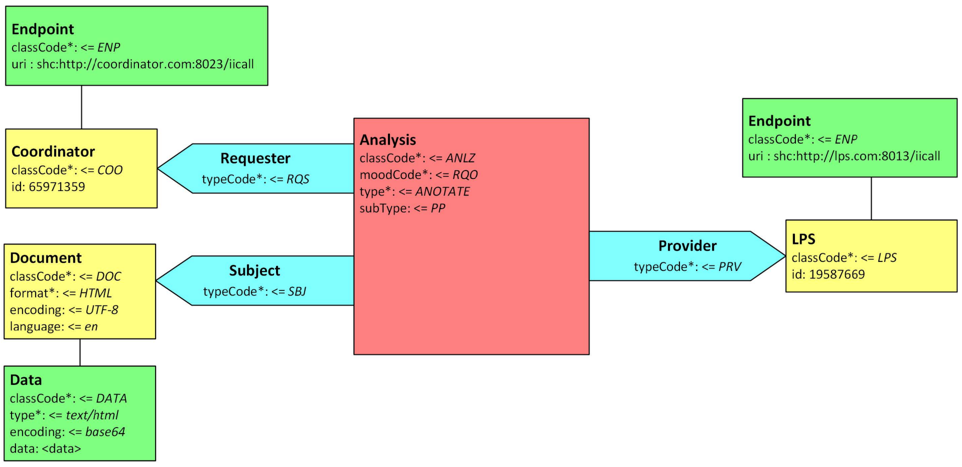


FIGURE 79. iiCALL Message from the Coordinator to the LPS

Here, an analysis-act is the central element. The `moodCode` expresses that this act shall be a request (`RQO`). The LPS is intended to provide an analysis in the form of annotated corpora (attribute `type = ANOTATE`). The `subType = PP` specifies that only prepositions are relevant for the analysis. The coordinator identifies itself by the endpoint, a Coordinator-Role, and the participation as a requester for this analysis (`typeCode = RQS`). The retrieved article serves as a subject for the analysis and takes part in the action as a Data-Entity in a Document-Role. Finally, the LPS is addressed via a concrete endpoint and specified as a provider for the analysis via an LPS-Role with corresponding Participation (`typeCode = PRV`).

The coordinator's exercise-handler passes the created analysis-act to the framework. In the case of the example, the URI of the endpoint of the provider is evaluated in order to forward the message to the corresponding connector and to transmit it to the addressed LPS. The framework at the LPS receives the message, extracts the act, and forwards it to the corresponding analysis-handler. Here, the analysis of the text is carried out using the parameters in the request using GATE. The result will be an annotated corpus, which is returned to the coordinator in a corresponding message. This message is shown in Figure 80 [**Galler, 2015**].

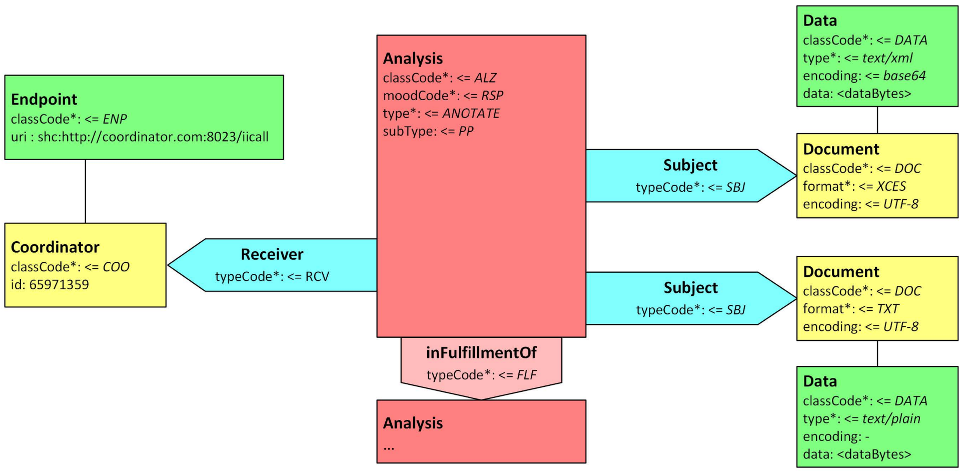


FIGURE 80. iiCALL Message from the LPS to the Coordinator

Compared to the previous message, the `moodCode` of the analysis-act has changed. In addition, the previously obtained request is concatenated in unchanged form via an `ActRelationhip` with the answer. The value `RSP` (as code for a response), together with the `typeCode = FLF` (code for "inFulfillmentOf"), expresses the concatenation as being the answer to a request. As a subject, the annotated corpus is assigned to the action. The coordinator with its corresponding end point takes part in the action as receiver.

The Act created by the analysis-handler is passed to the framework at the LPS. On the basis of the Receiver-Participation and the assigned end point, the response message can be sent to the coordinator. The coordinator can now use the annotated corpus to create the cloze test. Therefore, the HTML markup of the article is modified in such a way that all prepositions in the article text are replaced by drop-down lists. These lists contain a correct and three incorrect answers. Incorrect answers are selected randomly from a choice of other prepositions occurring in the article text. The resulting cloze test is attached as subject to a new exercise-act, which is shown in Figure 81 [**Galler, 2015**].

The changed markup is then packaged in a Data-Entity. The exercise-act is identified as a response (`moodCode = RSP`). In this case, the original request is not concatenated through an ActRelationship because the data is no longer relevant for the extension in the browser. The end user is

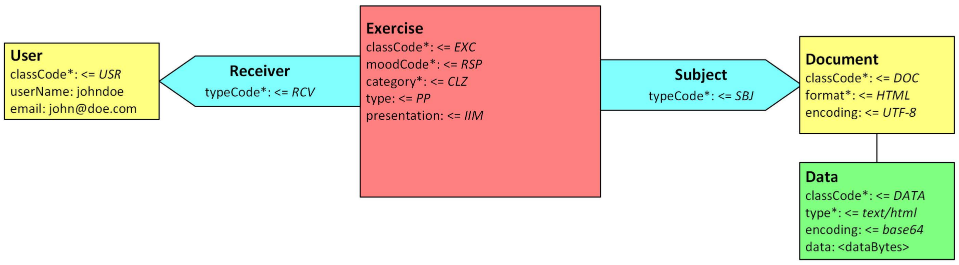


FIGURE 81. iiCALL Message from the Coordinator to the Client

the recipient of the message. The user will now receive the response from the coordinator. Based on the value of the attribute `presentation`, the extension in the browser recognizes the way in which the resulting cloze test can be presented, as shown on the right side of Figure 77. More details of implementation issues can be found in [**Galler, 2015**].

Part IV

Results and Prospects

CHAPTER 12

Evaluation

"I really like the idea of iiCALL. It's a very innovate approach to adapt HL7 V3's generic model and the development framework and transfer it to another domain! This proof of concept allows to think about further possibilities and might open a wide range of other application domains."

(Alexander Mense, Head of Technical Committee at HL7 Austria)

With the iiCALL Development Framework we defined a generic data model (i.e. the iiCALL GDM-Core) and a structured process how to technically implement learning scenarios. Therefore, based on the UML specification of a learning scenario, we refined the GDM and defined messages for data exchange and interaction between systems. To evaluate the iiCALL environment, we chose descriptive and qualitative methods. Thus, we reviewed features and requirements with respect to the prototype implementations which demonstrate the proof of concept for the iiCALL system.

12.1. Platform Independence

The server part of the prototype, which implements the iiCALL Software Framework and tools, is implemented in Java, so it is platform independent by default. The framework could also be realized in any other programming language, which on the one hand supports object-orientation to realize the specializations in the GDM, and which on the other hand supports the paradigm of reflection, and XML binding. Both are crucial to create and change classes during runtime. The client part of any iiCALL installation has to be implemented individually on working environments where the iiCALL system will be integrated. The client part of the demonstrated prototype is a Firefox plug-in which simply sends JSON messages to the server and processes responses. At the current stage of implementation, there is no framework for client development available. Within the prototype, client messages are generated based on predefined message templates.

12.2. Separation of Concerns

The described prototype shows a typical application in an iiCALL environment. Here, involved entities were divided into three roles, which are the user, the coordinator, and the LPS. It might be possible to combine the tasks of the coordinator and the LPS and to cover it in one system. However, this contradicts the paradigm of the Separation of Concerns and, in concrete terms, the goal of iiCALL to address different domain users. For instance, linguists and computer linguists can provide appropriate tools for linguistic analysis and transformation via an LPS. Educators and language teachers with their didactic know-how can contribute to the design of exercises, the tracking of the learning progress and the feedback to the user. Developers use the mechanisms in the coordinator at an abstract level to implement additional business logic. Developers of Web and mobile clients concentrate on the preparation of the content in terms of usability and user interface design.

12.3. Flexibility

The iiCALL Software Framework abstracts the processing of iiCALL messages and achieves flexibility through its modular design. From the perspective of developers, the entire logic of a service (with respect to an act) is implemented within an `ActHandler`. Messages are delivered in the form of acts in both directions, from and to the framework. This means for developers that they have to define proper R-MIM representations. Message exchange between heterogeneous systems is organized by the use of connectors. They are responsible for sending and receiving messages in a predefined manner and for transporting and serializing messages. Some specific connectors are already part of the iiCALL Software Framework, new connectors can be created by developers. When creating new connectors, one has to keep in mind to use unique short identifiers.

A major difference between the iiCALL development methodology and the methodology of the HL7 V3 standard is the way how the data model is implemented. In HL7 the data model is restricted and refined (see R-MIM and D-MIM). At the end of this process, a certain amount of message types exists for a specific domain. Those message types are generated from individual XML schemata depending on the used technology. In contrary, for the iiCALL environment, the goal is to map all relevant messages within one single XML schema based on the generic data model. This approach is intended to reduce the complexity of the messages and the complexity of implementations. It is an important aspect in the automatic mapping of the data model in the framework.

12.4. Extensibility

Extensibility is an important issue of iiCALL. Here, we give an overview how to get the source code of the iiCALL system, and with a simple example we demonstrate how to extend functionality by implementing a new iiCALL service.

Retrieving the iiCALL Source Code. The source code of the iiCALL environment is available for download from a GIT repository within the Bitbucket Web site[1]: Source code may be checked-out, which leads to a segmentation of the source code into five sub-projects (see Table 74). Currently, source code is organized in 61 classes within 8 packages [**Wahl et al., 2016**].

The Eclipse IDE for JEE developers having installed the Apache Ivy plug-in is the preferred development environment for iiCALL. After GIT

[1]https://bitbucket.org/iicalldev/iicall.git

TABLE 74. Sub-projects and Tasks of iiCALL Sources

Sub-project	**Task**
Client	Sources of a Firefox add-on showing client functionality
Coordinator	Sources of the coordinator; it is based upon the framework
Framework	Sources of the iiCALL framework
LPS	Language Processing Service (LPS), it is based upon the framework, it uses GATE
Website	Web site showing information of the prototype, including complete JavaDoc

check-out, LPS and coordinator are prepared to be started as Java Applications (see Figure 82 and Figure 83) [**Wahl et al., 2016**].

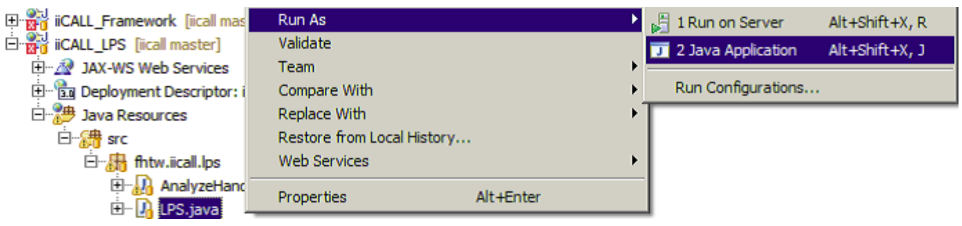


FIGURE 82. Starting the LPS

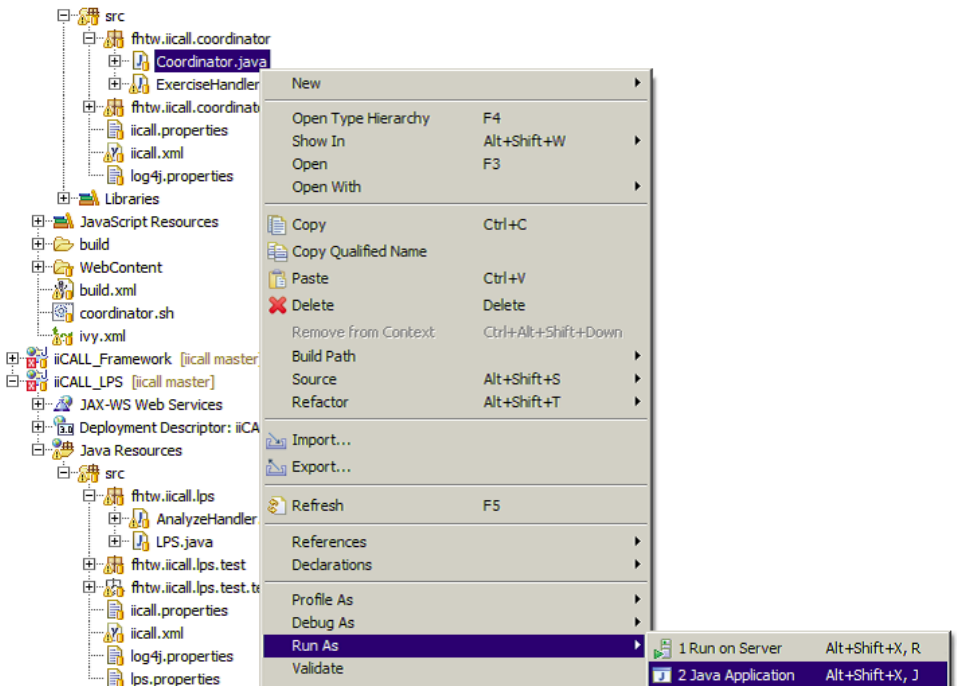


FIGURE 83. Starting the Coordinator

Implementing a New iiCALL Service. New iiCALL services can be implemented by starting with the existing LPS or coordinator project. Here, a new act-handler has to be created (see iiCALL framework overview in Figure 71 on page 130). It is possible to use Ant configuration and Ivy (build.xml and ivy.xml) for creating a new project, which both are included in the GIT version. To start the frameworks, one has to instantiate the class IICALL and then call the method `start()` of the created object. This can happen, for instance, inside the `main()` method. Recommended is the implementation as a daemon, so coordinator and LPS can be started at command line[2]. Otherwise, in case a new service is implemented by a new act-handler, the new mapping for the new act-handler must be defined by updating the iiCALL configuration file (see Listing 12 line 7 to 11). The `ActHandler` class is referenced by tag `<act-impl-class>`. It is associated with the tag `<act-code>`, which identifies the classCode-value of the GDM [**Wahl et al., 2016**].

```
1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <iicall-conf xmlns="http://technikum-wien.at/iicall/
       framework/conf"
4   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5   xsi:schemaLocation="http://technikum-wien.at/iicall/
       framework/conf iiCALLConf.xsd">
6
7   <act-mapping>
8    <act-code>EXC</act-code>
9    <act-impl-class>fhtw.iicall.coordinator.ExerciseHandler
10   </act-impl-class>
11  </act-mapping>
12
13  <connectors>
14   <connector-impl-class>fhtw.iicall.framework.connector.
       HTTPConnector
15   </connector-impl-class>
16  </connectors>
17
18 </iicall-conf>
```

Listing 12. Mapping New Act-handler

The class `GenericActHandler` of the framework must be extended by the `ActHandler` which then must implement the methods `start()`, `stop()`, and `handle()` of the interface `ActHandlerInterface`. Listing 13 shows the implementation for an exercise-act. Here, the method `start()` starts an HTTP client for calling Web sites [**Wahl et al., 2016**].

```java
public class ExerciseHandler extends GenericActHandler {

 protected Logger log = LogManager.getLogger(getClass());
 private static HttpClient httpClient = null;
 private static SecureRandom rand;
```

[2]see `http://commons.apache.org/proper/commons-daemon`

```

 @Override
 public void start() throws Exception {
  log.info("Starting http client...");
  httpClient = new HttpClient();
  httpClient.start();

  rand = SecureRandom.getInstance("SHA1PRNG");
 }

 @Override
 public void stop() throws Exception {
  httpClient.stop();
 }
```

Listing 13. Act-handler Code of Exercise-act

The method `handle()` must implement the logic of the act. The framework works on the GDM-Core level, so the parameter of the method must be of type `Act`. The configuration ensures that the correct type is passed. Nevertheless, lines 3–5 in Listing 14 explicitly checks the correctness.

```
1  @Override
2  public Act handle(Act act) throws Exception {
3   if (!(act instanceof Exercise) || act.getClassCode() !=
      EActClass.EXC) {
4    throw new ActHandlerException("Act-Subclass 'Exercise'
      expected");
5   }
6
7   Exercise exercise = (Exercise) act;
8   Exercise response = null;
9
10  EActMood moodCode = exercise.getMoodCode();
11  if (moodCode == EActMood.RQO) {
12   response = handleRequestOrder(exercise);
13  } else if (moodCode == EActMood.RSP) {
14   // TODO
15  } else {
16   throw new ActHandlerException("Unsupported moodCode: " +
      moodCode.toString());
17  }
18
19  return response;
20 }
```

Listing 14. Act-handler Code handle() Method

Lines 11–17 show, how handling will be processed related to the kind of act. The `moodCode` decides, if it is a request (`RQO`) or a response (`RSP`). For instance, a request is then processed by the method `handleRequestOrder()`. Finally, the method returns the resulting response-act to the framework (see lines 12 and 19) [**Wahl et al., 2016**].

Creation and Passing of Acts. The following Listing 15 expresses a method `createAnalysisAct()`. It creates an analysis-act which shall request an analysis within an `ExerciseHandler`. Therefore, one has to create an instance of `fhtw.iicall.framework.model.ObjectFactory` (see line 4). Then, the needed elements and attributes are defined in lines 4–34. In lines 37–53, all elements are joined and each element is assigned the corresponding `classCode` using method `setClassCode()` [**Wahl et al., 2016**].

```
 1 public static Analysis createAnalysisAct() {
 2  ObjectFactory factory = new ObjectFactory();
 3
 4  Analysis als = factory.createAnalysis();
 5  als.setClassCode(EActClass.ALZ);
 6  als.setMoodCode(EActMood.RQO);
 7  als.setType(EAnlType.ANOTATE);
 8  als.setSubType(EAnlSubType.ADVP);
 9
10  Coordinator coordinator = factory.createCoordinator();
11  coordinator.setClassCode(ERoleClass.COO);
12
13  Endpoint coordinatorEP = factory.createEndpoint();
14  coordinatorEP.setClassCode(EEntityClass.ENP);
15  coordinatorEP.setUri(COORDINATOR_EP);
16
17  LPS lps = factory.createLPS();
18  lps.setClassCode(ERoleClass.LPS);
19
20  Endpoint lpsEP = factory.createEndpoint();
21  lpsEP.setClassCode(EEntityClass.ENP);
22  lpsEP.setUri(LPS_EP);
23
24  Document doc = factory.createDocument();
25  doc.setClassCode(ERoleClass.DOC);
26  doc.setFormat(EDocFormat.HTML);
27  doc.setEncoding(EDocEncoding.UTF_8);
28  doc.setLanguage("de");
29
30  Data data = factory.createData();
31  data.setClassCode(EEntityClass.DAT);
32  data.setEncoding(EDataEncoding.BASE_64);
33  data.setType(EDataType.TEXT_XML);
34  data.setData(CorpusUtils.encodeBase64("some text encodede
       her"));
35
36  // assemble message
37  Participation requester = factory.createParticipation();
38  requester.setTypeCode(EParticipationType.RQS);
39  als.getParticipants().add(requester);
40  requester.setRole(coordinator);
41  coordinator.setEntity(coordinatorEP);
42
43  Participation subject = factory.createParticipation();
44  subject.setTypeCode(EParticipationType.SBJ);
```

```
    als.getParticipants().add(subject);
    subject.setRole(doc);
    doc.setEntity(data);

    Participation receiver = factory.createParticipation();
    receiver.setTypeCode(EParticipationType.RCV);
    als.getParticipants().add(receiver);
    receiver.setRole(lps);
    lps.setEntity(lpsEP);

    als.setId(als.getUID());

    return als;
}
```

Listing 15. Method createAnalysisAct()

Listing 16 shows part of the code from an exercise-handler. Preparation for the analysis request is created by `createAnalysisAct()` method. The method `findEntity()` identifies the Entity object that is assigned the `classCode DAT`. The text, that is meant to be analyzed, is assigned to this object (lines 5–9). Finally, in line 11, the framework receives the prepared Act through a Connector using the `forwardAct()` method. At client side, it remains to visualize the analyzed text in correlation with the learning scenario [**Wahl et al., 2016**].

```
1  // assemble analysis act, attach article and forward to LPS
2  Analysis analysis = TestMsgFactory.createAnalysisAct();
3  Data data = (Data) analysis.findEntity(EEntityClass.DAT);
4
5  data.setEncoding(EDataEncoding.BASE_64);
6  data.setData(CorpusUtils.encodeBase64(article.toString()));
7  data.setType(EDataType.TEXT_HTML);
8  analysis.setType(EAnlType.ANOTATE);
9  analysis.setSubType(EAnlSubType.valueOf(((Exercise) exercise
       ).getType().toString()));
10
11 Act analysisResponse = forwardAct(analysis);
```

Listing 16. Method exerciseHandler()

The source code comes with source code comments ready to create full source code documentation using JavaDoc. A list of packages and corresponding classes taken from the JavaDoc is shown in Appendix C on page 182.

Code Reuse. The iiCALL development allows collaborative development. The current prototype provides several utilities to be re-used. For instance, the class Corpus provides general methods to transform annotated corpora in various formats, the class Serialization provides different methods to serialize messages, and the class XMLValidationHandler provides utilities to log errors and warnings during validation of XML documents. JavaDoc is available for the entire source code to get a faster overview of re-usable functionalities.

12.5. Requirements Review

Being an ICALL System. An ICALL system is a CALL system that allows the usage of NLP technologies. Therefore, the proposed iiCALL environment fulfills the requirement of being an ICALL system because it integrates GATE in its system architecture. GATE on the one hand provides ready-to-use NLP functions and on the other hand, it allows to extend its features by adding individual functions.

Integration in Common Working Environments. Integration in common working environments like Web browsers or email clients was exemplary shown by the two prototypes, the context-related vocabulary trainer and the prototype implementing the automatic exercise learning scenario. Integration works on such environments if it is possible to extend their functionalities by an individual software, be it by an add-on or plug-in. Additionally, the environment must allow to access the current content of the environment (to be used as content for language learning) and to send messages to the iiCALL server and process received messages from the server. For instance, Web browsers like Firefox, Chrome or Opera allow such implementation, email clients like Thunderbird do as well. Therefore, iiCALL meets the requirement of integration.

Support of Arbitrary Languages. Language learning shall not be limited to a subset of natural languages. Technical limitations may occur if, for instance, specific characters or punctuation marks cannot be visualized due to a lack of proper encoding standards. The iiCALL system fully supports the Unicode standard, so this requirement can be seen as fulfilled. Developers of new iiCALL clients have to ensure that Unicode is supported by the working environments in which the client will be integrated.

Multimedia Support in Learning. Multimedia support was not part of the shown prototypes, so it could not be examined. Technically, the iiCALL system is prepared for multimedia as message exchange can use the MIME standard to transfer multimedia content.

Support of Different Learning Scenarios. In principle, a learning scenario can be defined considering the learning content, involved users and entities, and the sequence of learning process steps. The iiCALL Development Framework is a structured method to define learning scenarios starting with UML use case specifications. Technically, after specifying the use cases the mapping to the GDM is defined and messages for data exchange are created. Here, learning workflows are represented by the application of the GDM which allows to define actions, flows of actions, and involved entities (e.g. users, content, etc.) taking part in such actions within specific roles. A proof of concept is given by the prototype of the automatic exercise learning scenario.

Context-related Language Learning. The context-related vocabulary trainer showed that referring to a specific context in learning can be implemented by extending GATE with a new functionality. Actually, the introduced new GATE functionality can identify the domain of the learning text (medicine, IT, or sports). However, relevant context for language learning is not limited to the domain of content, it must be seen from a broader point of view, e.g. by considering circumstances and related situations in which words and phrases are used. The integration of semantic information can fulfill such a requirement. GATE allows to integrate semantic data as this was one of the requirements for the evaluation of proper NLP toolkits. We did not show this potential in the iiCALL prototypes, and we leave this to future research.

12.6. Scalability

In the automatic exercise learning scenario, only synchronous communication is modeled. In the sequence diagram (see Figure 68 on page 124) only synchronous message exchange happens between coordinator and LPS. However, several language analyses or transformations may be necessary for the preparation of an exercise. In such case, the coordinator will possibly need responses of various LPS. If these requests do not depend on each other, they need to be parallelized. In this case, an asynchronous communication process between the coordinator and LPS is desirable in order to keep the response time as low as possible for the user. Scalability may be seen with respect to system scalability. An increase of iiCALL users and an increase of concurrent usage of iiCALL services will bring with them additional challenges which were not discussed within the prototypes and also left to future work.

12.7. Research Questions and Hypotheses

To summarize, we can now answer the mentioned research questions and confirm the corresponding hypotheses.

Research Question 1. *How can a system architecture, an underlying data model, and a technology-oriented development process look like to fulfill the requirements of an Intelligent Integrated Computer-Assisted Language Learning environment?*

We introduced the underlying data model as the iiCALL GDM-Core and the corresponding refinements. The prototype of the automatic exercise learning scenario is based on the iiCALL GDM and proves the concept of the GDM. Figure 70 on page 128 shows the general iiCALL architecture including clients and a software framework which the system is based on. GATE is integrated at the server part of the iiCALL architecture and allows the application of NLP. The system follows a service-oriented architecture. It sends messages (currently serialized by JSON or XML) between components of the system. The proposed iiCALL Development Framework (see Chapter 8 on page 69) expresses a method to specify those messages. It starts with use case specification using UML, and then, by mapping and refining the GDM, it guides how to specify messages to be used for data exchange between the iiCALL components. The prototype explains how integration in working environments is possible and how learning scenarios can be designed and implemented. The introduced iiCALL Software Framework is the basis for further iiCALL implementations. It offers methods for communication and data exchange between the iiCALL components, it implements features to apply the GDM, and it enables to integrate GATE and NLP functionalities. The operational capability of the framework, which is shown to be extendable, therefore also confirms HYPOTHESIS 1:

HYPOTHESIS 1. *An extendable software framework, which can integrate Natural Language Processing functionalities, and which is based on a generic data model allows the implementation of an Intelligent Integrated Computer-Assisted Language Learning system.*

RESEARCH QUESTION 2. *Is there a comparable model that can be adapted to create the Intelligent Integrated Computer-Assisted Language Learning system?*

We found a comparable model, not related to functionalities but related to the technical architecture and operations, in the healthcare domain by the HL7 V3 standard. There, we identified similarities in the technical requirements and adapted them to the language learning domain. Our iiCALL Development Framework is a modification of the HL7 HDF, and our iiCALL GDM is comparable to the HL7 V3 Reference Information Model Backbone. Both are used for the development and implementation of the iiCALL prototype. Hence, also HYPOTHESIS 2 can be confirmed:

HYPOTHESIS 2. *Due to similarities to the health domain, the ideas of the Health Level Seven International can be adapted for the Intelligent Integrated Computer-Assisted Language Learning development.*

CHAPTER 13

The iiCALL Development and its Educational Impacts

The initial focus of our research on iiCALL was the design of a proper development method and the implementation of a software framework to fulfill the requirements of an iiCALL environment. With the introduction of the iiCALL Development Framework Version 1 and the iiCALL Software Framework, we included our research results in academic education at the University of Applied Sciences Technikum Wien. Research-oriented teaching methodology plays a significant role in academic education, especially in master level degree programs. Development of the iiCALL environment entails the use of multiple competencies in the field of Information Technologies including but not limited to Natural Language Processing, software programming and software architecture design, along with the skills on project management and the ability to work in a team. Here, we describe the experiences of integrating research in academic education. Also, we outline the effect on courses designed by competence-oriented learning outcomes [**Wahl and Winiwarter, 2013a**].

13.1. Research-based Teaching

The application of research-based teaching methodology is based on concepts of [**Fink, 2003**] which have demonstrated vital positive results in the learning process. The research-based learning process commences with laying down learning goals (these are based on the specific research issues) followed by the definition of teaching and learning activities. To ensure that the students can adjust their learning attempts, a feedback loop of periodic evaluation and assessment is put in place, as illustrated in Figure 84. The primary objective of research-based teaching is to produce student-centered learning results, which implies acquiring knowledge and enhancing proficiency and skills in numerous disciplines. In some situations, particularly when the students with specific prior knowledge are concerned, research-based teaching methodology also helps the research progress.

13.2. Learning Goals for iiCALL

We need to dive deeper into the system overview of iiCALL to clearly demonstrate the learning goals in the iiCALL research-based learning. Web service technologies are implemented to allow browser plug-ins to send communication to and forth the learning server. At the server end, the core software framework controls the data exchange and is responsible for persistence. NLP tasks are performed by GATE, which is a framework completely realized in Java and which comes with previously implemented NLP features.

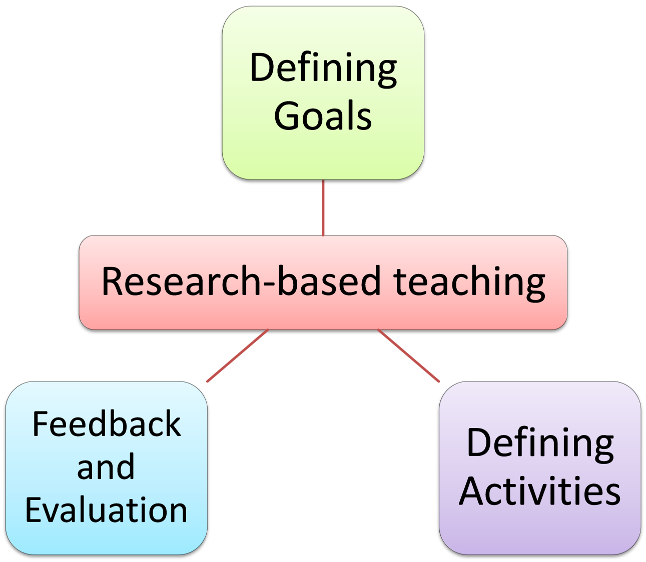


FIGURE 84. Research-based Teaching for Significant Learning Outcome

Based on the functional requirements of iiCALL, Table 75 presents the primary learning goals and the corresponding competencies and skills that are required by the students to achieve their goals.

TABLE 75. The Learning Goals and Competences of Research-based Teaching Using iiCALL

Learning Goals	**Competences and Skills**
Requirements engineering	Capability of problem analysis; Use case identification; Use case specifications; Applying the Unified Modeling Language (UML)
Software architecture	Knowledge of client-server architecture; Knowledge of n-tier architecture; Interface specifications; Data modeling
Prototyping	Implementation of Browser plug-ins; Integration of software framework; Implementation of Web services; Integration of persistence frameworks; NLP algorithms
Standards	Knowledge about NLP standards; Knowledge about e-learning standards
Project management	Project leading; Team work

13.3. Learning Activities for iiCALL

Learning goals and learning activities are interconnected. The activities use different methods, beginning with literature research, analysis of the problem, documentation, and prototypical implementation. The entire process of learning is principally conformed to the idea of Problem-based Learning (PBL). The central issue of fulfilling the goals of iiCALL is to split it into sub-problems which the students are directed to solve in teams. We initiate the inclusion of iiCALL into education by examining the concept of iiCALL. Initial exercises relate to requirements engineering. Teachers (in the role of coaches) act as "customers" to simulate a customer and contractor situation here. Students' work yields to functional and non-functional requirements. These results help in identifying corresponding use cases which are specified using the UML standard for future implementations. This constitutes use case diagrams, sequence diagrams, and activity diagrams to support the software development process. Related use cases in iiCALL are classified in Table 76.

Table 76. Use Case Categories and Exemplary Use Cases

Use case category	**Exemplary use cases**
User profiles	Create account; Login; Change profile data; Change learning needs; Change language preferences;
Language Testing	Do language screening test; Do vocabulary trainer; Do cloze test; Do social translation; Do grading test;
User progress	Check personal language skills level; Synchronize with server; Re-do test; View personal statistics
Communication	Ask for help; Communicate using social media platform; Contact native speaker; Open chat;

Based on the specifications of use cases, system needs had to be defined. The students were given an appropriate infrastructure. Then, the implementation and design of the software architecture for iiCALL had to be performed by the students. Hence, various options were selected and assessed. Client-server architecture and 3-tier architecture along with the interface specifications were major parts of teaching. At the client end, software prototyping produced a functioning plug-in for the Mozilla Firefox browser. This plug-in implemented the use case "Do vocabulary trainer" which provides context-related language learning [**Wahl and Winiwarter, 2012a**]. Standards are significant factors related to the iiCALL development, on the one hand e-learning standards and on the other hand standards of NLP. Following the rules to work in teams, the students did comply by project management rules. Students were provided an option to choose between different project management models, like PMI or even Scrum, and a comprehensive project documentation was expected at the end.

13.4. Feedback and Evaluation of iiCALL

The role of teachers in research-based learning is comparable with the role of teachers in PBL. Teachers act as coaches and aid the students to discover the ways to solve a specific problem. In conventional PBL, the teacher is aware of methods of solving a particular problem. If PBL is practiced in research, it is essential that the students communicate and associate with people working on that research topic. Alternatively, the coaches are required to actively contribute to the research as well [**Wahl et al., 2009**]. In the iiCALL research-oriented teaching, all coaches are accustomed to the present state of research. Feedback is provided on the applicability of the research in iiCALL at the end. Discussions with coaches are held frequently which encourages students to learn the integration in iiCALL and improve their research actively.

13.5. Effect on Academic Education and on Progress of iiCALL

From the student's perspective, research-oriented teaching is advantageous because the content of learning has practical relevance. Furthermore, the iiCALL research offers multiple diverse IT-related sub-problems that the students can crack by teaming up with other students. Project management methods and techniques are put to use here, and the role of coaches is taken up by respective teachers. Coaches are required to aid and assist students in the problem-solving process. Since research innovation cannot ensure that a solution exists or is known, research-based learning is fitting for students with an advanced level of knowledge and previous experience with the technology. Therefore, research-based learning is perfect for students admitted to a master level course.

The initial effort to define the research idea increases with the complexity of the research. In iiCALL, language learning, NLP, and semantics of languages have to be explained to IT students, while students already know technical IT basics. The challenge of iiCALL is to understand how context-related language learning and the abstraction of learning processes can be transferred into IT. It was observed that working on an iiCALL environment boosted the motivation of the students. Research demonstrates the utility of real world applications and draws them closer to the idea of innovation. Since most students lack major practical experience, the inputs and ideas from them seldom support the research progress directly. Coaches are required to filter the input from the students and recognize valuable sub-results.

CHAPTER 14

Conclusion and Further Steps

The key contributions of this thesis can be summarized by contributions to Language Learning Software Development, and contributions to Academic Education. Firstly, we presented in this thesis a software framework, a generic data model, and a development method for implementing an integrated language learning platform. Secondly, we showed how we integrate the research activities and further developments in academic education at the University of Applied Sciences Technikum Wien.

In our research, "integrated" means, on the one hand, that learning should be possible within specific working environments. On the other hand, we define "integrated" within a broader perspective. We want to include the whole development process which starts from requirements analysis, and results in a standardized technology-oriented implementation. We define a platform that fulfills our extended definition of "integration": an Intelligent Integrated Computer-Assisted Language Learning (iiCALL) platform. A core requirement of our presented language learning platform is being "intelligent", which means that NLP technologies may be used in the processing of the languages. We compared suitable NLP tools to be integrated in our language learning system architecture. Language learning is highly complex because of the large number of possible languages and possible learning scenarios. For implementation purposes, we identified the need for a generic and flexible data model as well as a proper development process. A review of the healthcare domain and corresponding software systems directed us to models and methods used in the HL7 V3 standard. We evaluated the HL7 standard with regard to a possible adaptation to the development of the iiCALL system. Based on this, through the evolution of the data model, the versions of development methods and prototypes, we showed the proof of concept that the introduced Generic Data Model (GDM) and the iiCALL Development Framework lead to a flexible and expandable environment which allows language learning according to our definition. With the iiCALL Development Process and the iiCALL Generic Data Model it is possible to develop and implement language learning scenarios in a standardized manner. For this, the GDM must be refined based on functional requirements, and implementation is done by extending the iiCALL Software Framework.

Our research and development of the iiCALL system was continuously integrated in academic education at the University of Applied Sciences Technikum Wien. Within specific courses especially in the field of Software Engineering, the iiCALL system is used to teach software programming, Web programming, and project management. In addition, different master and bachelor theses deal with specific NLP or language learning issues from a scientific point of view.

14.1. Further Steps

The current state of development of our iiCALL environment builds a solid foundation for further research and developments. There remain some issues to be addressed. We plan to implement further language learning scenarios which implies the need for dealing with semantic data integration and usability. Additionally, with the growth of the system and the increase of users, we will have to think of proper scalability and security solutions.

Achieving Language Learning Potential. At the current stage of development, the iiCALL system works fine to be used for educational purposes, less for the goal to teach and learn natural languages, but for the goal to teach Software Engineering, software programming, and project management. Therefore, the current version of the iiCALL environment is used within several IT courses and practical works at the University of Applied Sciences to extend the functionalities by adding additional use cases. Adding several typical use cases, i.e. implementing learning scenarios are one crucial part to reach the goal for iiCALL of being a real language learning platform. A need of extending the iiCALL Software Framework by utilities may occur as a result of requirements engineering for further learning scenarios. At the client side, improvement will be done by implementing the iiCALL framework to be used in client development. Till now, messages sent from the client are restricted to message templates but are not generated generically.

Scalability. Our research was not focused on the scalability of the iiCALL system. Scalability with regards to the possibility of adding new functionalities is given by default by applying the introduced development process and the strict usage of the iiCALL GDM. When we have a closer look at the prototype based on Figure 68, we only see a synchronous communication sequence for the query between the coordinator and the LPS. As already mentioned, more than one LPS may be necessary for the preparation of a specific learning scenario. If such requests do not depend on each other, they have to be parallelized. In this case, we have to address asynchronous communication issues.

From the perspective of system scalability, i.e. the server setup and performance issues, there remain things to be done, which will be handled mostly by system and database administrators.

Usability. For the proposed iiCALL prototypes there was little effort invested for user interface design and usability as it was not our focus of research. Further developments will follow the principles of User-Centered Design (UCD). UCD is applied before developers implement the final user interface, furthermore, it is done at the beginning of the design phase. There are different ways to identify usability issues, e.g. by providing mock-ups or by designing user interface prototypes to test users and, therefore, identify user interface needs.

At the University of Applied Sciences Technikum Wien, we can test user interfaces in our dedicated Usability Laboratory. There, different eye-tracking devices with a corresponding analysis software are accessible. Those devices are used to monitor user behavior. We evaluate user interactions in a way that we provide specific tasks to test users. For the iiCALL system those tasks are, e.g. “Change your user preferences!” or “Start and pass a preposition cloze test!”. For each task, the eye-tracking device can identify user interactions (where the user looks or where he clicks) while the user tries to fulfill the task. The analysis software evaluates the user interactions, which we can use to identify difficulties and necessary changes in the user interface design.

Semantic Data Integration. The inclusion of semantic data in language learning processes can be used, e.g. for reasoning purposes, automatic summarization of texts, automatic generation of questions and answers (Q&A) games, etc. The iiCALL system is prepared to integrate semantic data because of GATE’s feature to interface such data using different data formats. Relevant semantic data is available from different platforms like the Linked Data[1] platform or the Linguistic Linked Open Data[2] platform.

Security. For the future scenario to open the iiCALL system for the broad public, finally, we have to consider security issues, be it on the client side as well as on the server side. Secure data transfer and secure storage of user data are as important as the operation on a hardened server operating system. To deal with Web Application Security, we will follow the Open Web Application Security Project (OWASP)[3] guidelines.

[1]`http://linkeddata.org`
[2]`http://linguistic-lod.org/`
[3]`https://www.owasp.org`

List of Tables

List of Figures

List of Abbreviations

ACE	Automatic Content Extraction 20
ACL	Association for Computational Linguistics 8
AI	Artificial Intelligence 32
AJAX	Asynchronous JavaScript and XML 61
ANSI	American National Standards Institute 45
ATG	Automatic Test Generation 60
CALICO	Computer-Assisted Language Instruction Consortium 8
CALL	Computer-Assisted Language Learning ii
CDA	Clinical Document Architecture 45
CEFR	Common European Reference Framework for Languages . 9
CEN	European Committee for Standardization 23
CIM	Constrained Information Model
CLT	Cognitive Load Theory 14
D-MIM	Domain Message Information Model 47
DCA	Dual Channel Assumption 15
DIN	Deutsches Institut für Normung 23
EAB	Educational Activities Board 24
ECL	Extraneous Cognitive Load 14
EUROCALL	European Association for Computer-Assisted Language Learning 8
GATE	General Architecture for Text Engineering 35
GCL	Germane Cognitive Load 14
GCM	Generic Component Model 43
GDM	Generic Data Model 6
HDF	Healthcare Development Framework ii
HL7	Health Level Seven International ii
HMD	Hierarchical Message Description 47
ICALL	Intelligent Computer-Assisted Language Learning ii
ICL	Intrinsic Cognitive Load 14
IEC	International Electrotechnical Commission 23
IEEE	Institute of Electrical and Electronics Engineers 23
iiCALL	Intelligent Integrated Computer-Assisted Language Learning ii
ILTS	Intelligent Language Tutoring Systems 60
IMS	IMS Global Learning Consortium 23
IMS CP	IMS Content Packaging 24
INLOC	Integrating Learning Outcomes and Competences 25
IR	Information Retrieval 2
ISCA	International Speech and Communication Association 8
ISO	International Organization for Standardization 18

ITS	Implementation Technology Specifications	50
JAXB	Java Architecture for XML Binding	129
LE	Language Engineering	36
LMF	Lexical Markup Framework	18
LMS	Learning Management Systems	26
LOM	Learning Object Metadata	24
LPS	Language Processing Service	125
LR	Lexical Resources	18
LTSC	Learning Technology Standards Committee	27
MDA	Model Driven Architecture	27
MIME	Multipurpose Internet Mail Extensions	135
MRD	Machine Readable Dictionary	18
MT	Machine Translation	32
NA	Network Analysis	37
NER	Named Entity Recognition	34
NLP	Natural Language Processing	ii
NLTK	Natural Language ToolKit	35
OWASP	Open Web Application Security Project	158
OWL	Web Ontology Language	39
PBL	Problem-based Learning	154
POS	Part of Speech	34
R-MIM	Refined Message Information Model	47
RIM	Reference Information Model	ii
RMRS	Robust Minimal Recursion Semantics	37
SCORM	Sharable Content Object Reference Model	26
SOA	Service-Oriented Architecture	27
SPL	Structured Product Labeling	45
TFS	Typed Feature Structure	18
UCD	User-Centered Design	157
UML	Unified Modeling Language	18
UMLS	Unified Medical Language System	39
URI	Uniform Resource Identifier	125
VIEW	Visual Input Enhancement of the Web	10
W3C	World Wide Web Consortium	27
WERTi	Working with English Real Texts	11
WFMS	Workflow Management System	27
xAPI	Tin Can API	27
XCES	Corpus Encoding Standard for XML	134
XJC	XML Java Compiler	129
XUL	XML User Interface Language	61

Bibliography

[Amaral and Meurers, 2011] Amaral, L. and Meurers, D. (2011). On Using Intelligent Computer-Assisted Language Learning in Real-Life Foreign Language Teaching and Learning. *ReCALL*, 23(1):4–24.

[Antoniadis et al., 2005] Antoniadis, G., Echinard, S., Kraif, O., Lebarbé, T., Loiseau, M., and Ponton, C. (2005). Mirto: A New Approach for Call Systems. In Courtiat, J.-P., Davarakis, C., and Villemur, T., editors, *Technology Enhanced Learning: IFIP TC3 Technology Enhanced Learning Workshop (TeL'04), World Computer Congress, August 22–27, 2004, Toulouse, France*, pages 147–156. Springer US, Boston, MA.

[Antoniadis et al., 2013] Antoniadis, G., Granger, S., Kraif, O., Ponton, C., and Zampa, V. (2013). NLP and CALL: integration is working. *CoRR*, abs/1302.4814.

[Baddeley, 1992] Baddeley, A. (1992). Working memory. *Science*, 255(5044):556–559.

[Bannert, 2002] Bannert, M. (2002). Managing cognitive load–recent trends in cognitive load theory. *Learning and instruction*, 12(1):139–146.

[Bass et al., 2012] Bass, L., Clements, P., and Kazman, R. (2012). *Software Architecture in Practice.* Addison-Wesley Professional, 3rd edition.

[Blobel, 2011] Blobel, B. (2011). Intelligent security and privacy solutions for enabling personalized telepathology. *Diagnostic Pathology*, 6(1):S4.

[Blobel et al., 2006] Blobel, B., Engel, K., and Pharow, P. (2006). Semantic interoperability–HL7 Version 3 compared to advanced architecture standards. *Methods of Information in Medicine*, 45(4):343–353.

[Bohl et al., 2002] Bohl, O., Schellhase, J., Sengler, R., and Winand, U. (2002). The Sharable Content Object Reference Model (SCORM) - A Critical Review. In *Proceedings of the International Conference on Computers in Education*, ICCE '02, pages 950–, Washington, DC, USA. IEEE Computer Society.

[Bontcheva et al., 2002] Bontcheva, K., Cunningham, H., Tablan, V., Maynard, D., and Saggion, H. (2002). Developing Reusable and Robust Language Processing Components for Information Systems using GATE. In *3rd International Workshop on Natural Language and Information Systems (NLIS'2002)*, Aix-en-Provence, France. IEEE Computer Society Press.

[Boone, 2011] Boone, K. W. (2011). *The CDA^{TM} book.* Springer London.

[Bushehrian and Khaldar, 2011] Bushehrian, O. and Khaldar, R. (2011). An Extendable Software Architecture for Personalized E-Learning systems. *International Journal of Computer Applications*, 33(10):1–6.

[Buzzetto-More, 2006] Buzzetto-More, N. A. (2006). *Advanced Principles of Effective e-Learning.* Informing Science Press.

[Callmeier et al., 2004] Callmeier, U., Eisele, A., Schäfer, U., and Siegel, M. (2004). The Deep Thought Core Architecture Framework. In *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC) 2004*, pages 1205–1208, Lisbon, Portugal. ELRA, European Language Resources Association.

[Chandler and Sweller, 1992] Chandler, P. and Sweller, J. (1992). The split-attention effect as a factor in the design of instruction. *British Journal of Educational Psychology*, 62(2):233–246.

[Chi Michelene T. H., 1981] Chi Michelene T. H., Glaser Robert, R. E. (1981). *Expertise in problem solving.* Learning Research and Development Center, University of Pittsburgh [Pittsburgh, Pa.].

[Chua and Lee, 2009] Chua, F. and Lee, C. (2009). Collaborative learning using service-oriented architecture: A framework design. *Knowledge-Based Systems*, 22(4):271–274.

[Craik and Lockhart, 1972] Craik, F. I. and Lockhart, R. S. (1972). Levels of processing: A framework for memory research. *Journal of Verbal Learning and Verbal Behavior*, 11(6):671–684.

[Cunningham et al., 2014] Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V., Aswani, N., Roberts, I., Gorrell, G., Funk, A., Roberts, A., Damljanovic, D., Heitz, T., Greenwood, M. A., Saggion, H., Petrak, J., Li, Y., Peters, W., and Derczynski, L. (2014). Developing Language Processing Components with GATE Version 8. Technical report, University of Sheffield Department of Computer Science. Availabe (online): `https://gate.ac.uk/sale/tao/index.html` Accessed: 2017-01-06.

[Cunningham et al., 2013] Cunningham, H., Tablan, V., Roberts, A., and Bontcheva, K. (2013). Getting More Out of Biomedical Documents with GATE's Full Lifecycle Open Source Text Analytics. *PLOS Computational Biology*, 9(2):1–16.

[Dehbi et al., 2013] Dehbi, R., Talea, M., and Tragha, A. (2013). A Model Driven Methodology Approach for e-Learning Platform Development. *International Journal of Information and Education Technology*, 3(1):10–15.

[Dolin et al., 2006] Dolin, R. H., Alschuler, L., Boyer, S., Beebe, C., Behlen, F. M., Biron, P. V., and Shabo Shvo, A. (2006). HL7 Clinical Document Architecture, Release 2. *Journal of the American Medical Informatics Association : JAMIA*, 13(1):30–39.

[Domjan, 2009] Domjan, M. (2009). *The Principles of Learning and Behavior: Active Learning Edition*. Cengage Learning.

[Ehlers and Pawlowski, 2006] Ehlers, U.-D. and Pawlowski, M. J. (2006). *European Handbook of Quality and Standardisation in E-Learning*. Springer, Heidelberg.

[Ertmer and Newby, 1993] Ertmer, P. A. and Newby, T. J. (1993). Behaviorism, cognitivism, constructivism: Comparing critical features from an instructional design perspective. *Performance Improvement Quarterly*, 6(4):50–72.

[Fink, 2003] Fink, L. (2003). *Creating Significant Learning Experiences: An Integrated Approach to Designing College Courses*. Jossey-Bass higher and adult education series. Jossey-Bass higher and adult education series, Wiley.

[Francopoulo et al., 2007] Francopoulo, G., Bel, N., Georg, M., Calzolari, N., Monachini, M., Pet, M., and Soria, C. (2007). Lexical Markup Framework: ISO standard for semantic information in NLP lexicons. In *Workshop of the GLDV Working Group on Lexicography at the Biennial Spring Conference of the GLDV*, Tübingen, Germany.

[Francopoulo and George, 2008] Francopoulo, G. and George, M. (2008). Lexical Markup Framework (LMF). `http://www.lexicalmarkupframework.org/`. Accessed: 2017-01-06.

[Francopoulo et al., 2006] Francopoulo, G., George, M., Calzolari, N., Monachini, M., Bel, N., Pet, Y., and Soria, C. (2006). Lexical markup framework (LMF). In *Proceedings of LREC2006*.

[Freed and Borenstein, 1996a] Freed, N. and Borenstein, N. (1996a). RFC 2045: Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies. Technical report, Internet Engineering Task Force.

[Freed and Borenstein, 1996b] Freed, N. and Borenstein, N. (1996b). RFC 2046 (Draft Standard): Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types. Technical report, Internet Engineering Task Force. Updated by RFCs 2646, 3798, 5147.

[Galler, 2015] Galler, R. (2015). Framework für eine generische Sprachlernumgebung basierend auf iiCALL. Master's thesis, University of Applied Sciences Technikum Wien. Supervised by: Harald Wahl.

[Greene et al., 2004] Greene, C., Keogh, K., Koller, T., Wagner, J., Ward, M., and Van Genabith, J. (2004). Using NLP technology in CALL. In *Proceedings of InSTIL/ICALL 2004*.

[Havlicek, 2014] Havlicek, K. (2014). Implementing Client Part of Language Learning System. Master's thesis, University of Applied Sciences Technikum Wien. Supervised by: Harald Wahl.

[Heineman and Councill, 2001] Heineman, G. T. and Councill, W. T., editors (2001). *Component-based Software Engineering: Putting the Pieces Together*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

[Hinchley, 2007] Hinchley, A. (2007). *Understanding Version 3: A Primer on the HL7 Version 3 Healthcare Interoperability Standard - Normative Edition*. Understanding HL7 series. Alexander Mönch Publishing.

[HL7, 2010] HL7 (2010). HL7 Healthcare Development Framework (HDF). HL7 V3 Standard HDF Version 1.5, Release 1, Health Level Seven, Inc. `http://gforge.hl7.org/gf/download/frsrelease/608/6671/HDF_1.5.pdf` Accessed: 2017-01-06.

[HL7, 2016] HL7 (2016). HL7 Reference Information Model (RIM). HL7 V3 Standard RIM Version 2.47, Release 7, Health Level Seven International. `http://www.hl7.org/implement/standards/rim.cfm` Accessed: 2017-01-06.

[Hoel et al., 2010] Hoel, T., Hollins, P., and Pawlowski, J. (2010). On the Status of Learning Technology Specifications and Standards. *International Journal of IT Standards and Standardization Research*, 8(2).

[IEEE, 2002] IEEE (2002). IEEE Standard Learning Object Metadata. Standard, Institute of Electrical and Electronics Engineers.

[IEEE, 2005] IEEE (2005). IEEE Standard for Learning Technology – Extensible Markup Language (XML) Schema Definition Language Binding for Learning Object Metadata. Standard, Institute of Electrical and Electronics Engineers.

[Indurkhya and Damerau, 2010] Indurkhya, N. and Damerau, F. J. (2010). *Handbook of Natural Language Processing*. Chapman & Hall/CRC, 2nd edition.

[ISO, 2005a] ISO (2005a). ISO/IEC JTC 1/SC 36: Information technology – Learning, education and training - Quality management, assurance and metrics – Part 1: General approach. Standard, International Organization for Standardization.

[ISO, 2005b] ISO (2005b). ISO/TC 37/SC 4 WG1: Language Resource Management – Feature Structures – Part 1: Feature Structure Representation. Standard, International Organization for Standardization.

[ISO, 2008] ISO (2008). ISO/TC 37/SC 4 N453: Language Resource Management – Lexical Markup Framework (LMF). Standard, International Organization for Standardization.

[ISO, 2009a] ISO (2009a). ISO/IEC JTC 1 / SC 7: Information technology – Open distributed processing – Reference model: Architecture. Standard, International Organization for Standardization.

[ISO, 2009b] ISO (2009b). ISO/IEC JTC 1/SC 36: Information technology – Sharable Content Object Reference Model (SCORM^{TM}) 2004 3rd Edition – Part 2: Content Aggregation Model Version 1.1. Standard, International Organization for Standardization.

[Jones, 1994] Jones, K. S. (1994). Natural Language Processing: A Historical Review. In Zampolli, A., Calzolari, N., and Palmer, M., editors, *Current Issues in Computational Linguistics: In Honour of Don Walker*, pages 3–16. Springer Netherlands, Dordrecht.

[Kalyuga et al., 2003] Kalyuga, S., Ayres, P., Chandler, P., and Sweller, J. (2003). The expertise reversal effect. *Educational psychologist*, 38(1):23–31.

[Kawaguchi, 2008] Kawaguchi, K. (2008). JSR-000222 Java Architecture for XML Binding 2.2 - Maintenance Release 2. `https://jcp.org/aboutJava/communityprocess/mrel/jsr222/index2.html`. Accessed: 2017-01-06.

[Luzi et al., 2010] Luzi, D., Contenti, M., and Pecoraro, F. (2010). The HL7 RIM in the Design and Implementation of an Information System for Clinical Investigations on Medical Devices. In *E-Health: First IMIA/IFIP Joint Symposium, E-Health 2010, Held as Part of WCC 2010, Brisbane, Australia, September 20-23, 2010*, pages 5–14, Berlin, Heidelberg. Springer.

[Maraoui et al., 2012] Maraoui, M., Zrigui, M., and Antoniadis, G. (2012). Use of NLP Tools in CALL System for Arabic. *International Journal of Computer Processing of Oriental Languages*, 24(2):153–166.

[Mayer, 2009] Mayer, R. E. (2009). *Multimedia Learning*. Cambridge University Press, New York, NY, USA, 2nd edition.

[Mayer and Moreno, 2003] Mayer, R. E. and Moreno, R. (2003). Nine ways to reduce cognitive load in multimedia learning. *Educational psychologist*, 38(1):43–52.

[Meurers, 2013] Meurers, D. (2013). Natural Language Processing and Language Learning. In Chapelle, C. A., editor, *Encyclopedia of Applied Linguistics*, pages 4193–4205. Blackwell.

[Meurers et al., 2010] Meurers, D., Ziai, R., Amaral, L., Boyd, A., Dimitrov, A., Metcalf, V., and Ott, N. (2010). Enhancing Authentic Web Pages for Language Learners. In *Proceedings of the NAACL HLT 2010 Fifth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 10–18, Stroudsburg, PA, USA. Association for Computational Linguistics.

[MITRE, 2010] MITRE (2010). Annotation Scheme for Marking Spatial Expressions in Natural Language. Technical report, The MITRE Corporation.

[Peixoto et al., 2010] Peixoto, H., Machado, J., Neves, J., and Abelha, A. (2010). Semantic Interoperability and Health Records. In *E-Health: First IMIA/IFIP Joint Symposium, E-Health 2010, Held as Part of WCC 2010, Brisbane, Australia, September 20-23, 2010*, pages 236–237, Berlin, Heidelberg. Springer.

[Petasis et al., 2003] Petasis, G., Karkaletsis, V., Paliouras, G., and Spyropoulos, C. D. (2003). Using the Ellogon Natural Language Engineering Infrastructure. In *Proceedings of the Workshop on Balkan Language Resurces and Tools, 1st Balkan Conference in Informatics (BCI 2003)*, Thessaloniki, Greece.

[Piho et al., 2015] Piho, G., Tepandi, J., Thompson, D., Woerner, A., and Parman, M. (2015). Business Archetypes and Archetype Patterns from the HL7 RIM and openEHR RM Perspectives: Towards Interoperability and Evolution of Healthcare Models and Software Systems. *Procedia Computer Science*, 63:553–560.

[Ralph, 1998] Ralph, G. (1998). TIPSTER architecture design document version 3.1. Technical report, NIST - National Institute of Standards and Technology. Availabe (online): `http://www-nlpir.nist.gov/related_projects/tipster/` Accessed: 2017-01-06.

[Sag et al., 2003] Sag, I., Wasow, T., and Bender, E. (2003). *Syntactic Theory: A Formal Introduction.* CSLI lecture notes. Center for the Study of Language and Information.

[Schadow et al., 2006] Schadow, G., Mead, C. N., and Walker, D. M. (2006). The HL7 Reference Information Model Under Scrutiny. In Hasman, A., Haux, R., van der Lei, J., Clercq, E. D., and France, F. H. R., editors, *Medical Informatics Europe (MIE)*, volume 124 of *Studies in Health Technology and Informatics*, pages 151–156, Maastricht, Netherlands. IOS Press.

[Semple, 2000] Semple, A. (2000). Learning Theories and Their Influence on the Development and Use of Educational Technologies. *Australian Science Teachers' Journal*, 46(3):21–28.

[Smith and Ceusters, 2006] Smith, B. and Ceusters, W. (2006). HL7 RIM: an incoherent standard. *Studies in health technology and informatics*, 124(August):133–138.

[Sweller, 1988] Sweller, J. (1988). Cognitive load during problem solving: Effects on learning. *Cognitive Science*, 12(2):257–285.

[Sweller et al., 1990] Sweller, J., Chandler, P., Tierney, P., and Cooper, M. (1990). Cognitive load and selective attention as factors in the structuring of technical material. *Journal of Experimental Psychology: General*, 119:176–192.

[Sweller et al., 1998] Sweller, J., Merrienboer, J. J. G. V., and Paas, F. G. W. C. (1998). Cognitive architecture and instructional design. *Educational Psychology Review*, 10(3):251–296.

[Taus, 2013] Taus, M. (2013). Automatisierte Identifikation der deutschen Sprache durch morphologische Analyse - Sprachmodellentwicklung der literarischen Sprache im Bezug auf Konjunktion, Deklination und Komparation. Master's thesis, University of Applied Sciences Technikum Wien. Supervised by: Harald Wahl.

[Triesman and Davies, 1973] Triesman, A. and Davies, A. (1973). Divided Attention to eye and ear. In Kornblum, S., editor, *Attention and performance IV.*, pages 101–117. Academic Press, New York.

[van Merriënboer and Sweller, 2005] van Merriënboer, J. J. G. and Sweller, J. (2005). Cognitive Load Theory and Complex Learning: Recent Developments and Future Directions. *Educational Psychology Review*, 17(2):147–177.

[Wahl et al., 2015a] Wahl, H., Galler, R., and Winiwarter, W. (2015a). A Generic Software Framework for Intelligent Integrated Computer-Assisted Language Learning (iiCALL) Environment. In *Proceedings of the 14th International Conference on Web-based Learning (ICWL 2015)*, pages 264–270, Guangzhou, China. Springer International Publishing.

[Wahl et al., 2015b] Wahl, H., Galler, R., and Winiwarter, W. (2015b). Demonstration of the Generic Software Framework for the Integrated Intelligent Computer-Assisted Language Learning (iiCALL) Environment. In *Proceedings of the 17th International Conference on Information Integration and Web-based Applications & Services (iiWAS 2015)*, pages 450–455, Brussels, Belgium. ACM.

[Wahl et al., 2016] Wahl, H., Galler, R., and Winiwarter, W. (2016). Extending the Integrated Intelligent Computer-Assisted Language Learning (iiCALL) Environment. In *Proceedings of the 18th International Conference on Information Integration and Web-based Applications & Services (iiWAS 2016)*, pages 219–222, Singapore, Singapore. ACM.

[Wahl et al., 2009] Wahl, H., Mense, A., and Kaufmann, C. (2009). Can PBL be used for Knowledge Building in the HealthyIO Research Project? In *Proceedings of the 2nd International Research Symposium on Problem Based Learning (IRSPBL)*, Melbourne, Australia.

[Wahl and Winiwarter, 2011a] Wahl, H. and Winiwarter, W. (2011a). A Technological Overview of an Intelligent Integrated Computer-Assisted Language Learning (iiCALL) Environment. In *Proceedings of the World Conference on Educational Multimedia, Hypermedia and Telecommunications (EdMedia) 2011*, pages 3832–3837, Lisbon, Portugal. AACE.

[Wahl and Winiwarter, 2011b] Wahl, H. and Winiwarter, W. (2011b). The Intelligent Integrated Computer-Assisted Language Learning (iiCALL) Environment Work in Progress. In *Proceedings of the 13th International Conference on Information Integration and Web-based Applications & Services (iiWAS 2011)*, pages 426–429, Ho Chi Minh City, Vietnam. ACM.

[Wahl and Winiwarter, 2012a] Wahl, H. and Winiwarter, W. (2012a). A Context-Related Vocabulary Trainer in the Integrated Intelligent Computer-Assisted Language Learning (iiCALL) Environment. In *Proceedings of the 14th International Conference on Information Integration and Web-based Applications & Services (iiWAS 2012)*, pages 356–359, Bali, Indonesia. ACM.

[Wahl and Winiwarter, 2012b] Wahl, H. and Winiwarter, W. (2012b). A Prototypical Implementation of the Intelligent Integrated Computer-Assisted Language Learning (iiCALL) Environment. In *Proceedings of the 11th International Conference on Web-based Learning (ICWL 2012)*, pages 328–333, Sinaia, Romania. Springer.

[Wahl and Winiwarter, 2013a] Wahl, H. and Winiwarter, W. (2013a). Educational Impacts of the Intelligent Integrated Computer-Assisted Language Learning (iiCALL) Environment. In *Proceedings of the World Conference on Educational Multimedia, Hypermedia and Telecommunications (EdMedia) 2013*, pages 420–425, Victoria, Canada. AACE.

[Wahl and Winiwarter, 2013b] Wahl, H. and Winiwarter, W. (2013b). Towards a Generic Model for the Integrated Intelligent Computer-Assisted Language Learning (iiCALL) Environment. In *Proceedings of the 15th International Conference on Information Integration and Web-based Applications & Services (iiWAS 2013)*, pages 729–733, Vienna, Austria. ACM.

[Wahl and Winiwarter, 2014] Wahl, H. and Winiwarter, W. (2014). Exemplary Use Cases Based on the Generic Data Model for the Integrated Intelligent Computer-Assisted Language Learning (iiCALL) Environment. In *Proceedings of the 16th International Conference on Information Integration and Web-based Applications & Services (iiWAS 2014)*, pages 27–34, Hanoi, Vietnam. ACM.

[Wahl et al., 2010] Wahl, H., Winiwarter, W., and Quirchmayr, G. (2010). Natural Language Processing Technologies for Developing a Language Learning Environment. In *Proceedings of the 12th International Conference on Information Integration and Web-based Applications & Services (iiWAS 2010)*, pages 379–386, Paris, France. ACM.

[Wahl et al., 2011] Wahl, H., Winiwarter, W., and Quirchmayr, G. (2011). Towards an intelligent integrated language learning environment. *International Journal of Pervasive Computing and Communications*, 7(3):220–239.

[Watson, 1930] Watson, J. (1930). *Behaviorism.* Phoenix books. University of Chicago Press.

[Watson, 1913] Watson, J. B. (1913). Psychology as the Behaviorist Views it. *Psychological Review*, 20:158–177.

[Zhang et al., 2016] Zhang, Y.-F., Tian, Y., Zhou, T.-S., Araki, K., and Li, J.-S. (2016). Integrating HL7 RIM and ontology for unified knowledge and data representation in clinical decision support systems. *Computer Methods and Programs in Biomedicine*, 123:94–108.

[Zhou et al., 2008] Zhou, D., Zhang, Z., Zhong, S., and Xie, P. (2008). The Design of Software Architecture for E-Learning Platforms. In *Technologies for E-Learning and Digital Entertainment: Third International Conference, Edutainment 2008 Nanjing, China, June 25-27, 2008 Proceedings*, pages 32–40. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Zwickelbauer et al., 2015] Zwickelbauer, M., Merceron, A., Krauss, C., Kania, J.-P., and Scharp, M. (2015). Smart Learning: Der digitale Lernbegleiter für die berufliche Bildung. In Pongratz, H. and Keil, R., editors, *DeLFI 2015 - Die 13. E-Learning Fachtagung Informatik der Gesellschaft für Informatik e.V.*, pages 227–232. Bonner Köllen Verlag, Bonn.

APPENDIX A

The iiCALL Schema Definition

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<xs:schema version="1.0" xmlns:xs="http://www.w3.org/2001/XMLSchema"
 xmlns:re="http://technikum-wien.at/iicall/framework/model"
 targetNamespace="http://technikum-wien.at/iicall/framework/model"
 xmlns="http://technikum-wien.at/iicall/framework/model"
 elementFormDefault="qualified">

<!-- === Level 0 / BASE === -->

 <xs:complexType name="base">
  <xs:sequence />
 </xs:complexType>

<!-- === Level 1 / GDM-CORE === -->

 <!--  Act  -->
 <xs:complexType name="act">
  <xs:complexContent>
   <xs:extension base="base">
    <xs:sequence>
     <xs:element name="id" type="xs:ID" />
     <xs:element name="participants" type="participation" minOccurs="0" maxOccurs="unbounded" />
     <xs:element name="sourceActRel" type="actRelation" minOccurs="0" maxOccurs="unbounded" />
     <xs:element name="targetActRel" type="actRelation" minOccurs="0" maxOccurs="unbounded" />
    </xs:sequence>
    <xs:attribute name="classCode" type="eActClass" use="required" />
    <xs:attribute name="moodCode" type="eActMood" use="required" />
   </xs:extension>
  </xs:complexContent>
 </xs:complexType>

 <xs:simpleType name="eActClass">
  <xs:restriction base="xs:string">
   <xs:enumeration value="ACT" />
   <xs:enumeration value="EXC" />
   <xs:enumeration value="ALZ" />
 </xs:restriction>
</xs:simpleType>

<xs:simpleType name="eActMood">
 <xs:restriction base="xs:string">
  <xs:enumeration value="RQO" />
  <xs:enumeration value="RSP" />
 </xs:restriction>
</xs:simpleType>

<!--  ActRelation  -->
<xs:complexType name="actRelation">
 <xs:complexContent>
  <xs:extension base="base">
   <xs:sequence>
    <xs:element name="sourceAct" type="xs:IDREF" />
    <xs:element name="targetAct" type="xs:IDREF" />
   </xs:sequence>
   <xs:attribute name="typeCode" type="eActRelationType"
   use="required" />
  </xs:extension>
 </xs:complexContent>
</xs:complexType>

<xs:simpleType name="eActRelationType">
 <xs:restriction base="xs:string">
  <xs:enumeration value="PRC" />
 </xs:restriction>
</xs:simpleType>

<!-- ... Participation ... -->
<xs:complexType name="participation">
 <xs:complexContent>
  <xs:extension base="base">
   <xs:sequence>
    <xs:element name="role" type="role" minOccurs="1"
   maxOccurs="1" />
    <xs:element name="act" type="act" minOccurs="1"
   maxOccurs="1" />
   </xs:sequence>
   <xs:attribute name="typeCode" type="eParticipationType"
   use="required" />
  </xs:extension>
 </xs:complexContent>
</xs:complexType>

<xs:simpleType name="eParticipationType">
 <xs:restriction base="xs:string">
  <xs:enumeration value="RQS" />
  <xs:enumeration value="RCV" />
  <xs:enumeration value="PRV" />
  <xs:enumeration value="SBJ" />
 </xs:restriction>
</xs:simpleType>

<!--  Role  -->
<xs:complexType name="role">
 <xs:complexContent>
  <xs:extension base="base">
   <xs:sequence>
    <xs:element name="entity" type="entity" minOccurs="0"
   maxOccurs="1" />
    <xs:element name="participates" type="participation"
   nillable="true" minOccurs="0" maxOccurs="unbounded" />
   </xs:sequence>
   <xs:attribute name="classCode" type="eRoleClass" use="
   required" />
  </xs:extension>
 </xs:complexContent>
</xs:complexType>

<xs:simpleType name="eRoleClass">
 <xs:restriction base="xs:string">
  <xs:enumeration value="RLE" />
  <xs:enumeration value="USR" />
  <xs:enumeration value="DOC" />
  <xs:enumeration value="APR" />
  <xs:enumeration value="LPS" />
  <xs:enumeration value="COO" />
 </xs:restriction>
</xs:simpleType>

<!--  Entity  -->
<xs:complexType name="entity">
 <xs:complexContent>
  <xs:extension base="base">
   <xs:sequence>
   <xs:element name="servesAs" type="role" minOccurs="0"
   maxOccurs="unbounded" />
   </xs:sequence>
   <xs:attribute name="classCode" type="eEntityClass" use="
   required" />
  </xs:extension>
 </xs:complexContent>
</xs:complexType>

<xs:simpleType name="eEntityClass">
 <xs:restriction base="xs:string">
  <xs:enumeration value="ENT" />
  <xs:enumeration value="PSN" />
  <xs:enumeration value="ENP" />
  <xs:enumeration value="DAT" />
 </xs:restriction>
</xs:simpleType>
```

```

<!-- === Level 2 / GDM SPECIALIZED === -->

 <!--  Act  -->
 <xs:complexType name="exercise">
  <xs:complexContent>
   <xs:extension base="act">
    <xs:sequence>
     <xs:element name="category" type="eExcCat" minOccurs="1
    " maxOccurs="1" />
     <xs:element name="type" type="eExcType" minOccurs="0"
    maxOccurs="1"/>
     <xs:element name="presentation" type="eExcPres"
    minOccurs="0" maxOccurs="1"/>
    </xs:sequence>
   </xs:extension>
  </xs:complexContent>
 </xs:complexType>

 <xs:simpleType name="eExcCat">
  <xs:restriction base="xs:string">
   <xs:enumeration value="CLZ" />
   <xs:enumeration value="VOC" />
   <xs:enumeration value="QUZ" />
  </xs:restriction>
 </xs:simpleType>

 <xs:simpleType name="eExcType">
  <xs:restriction base="xs:string">
   <xs:enumeration value="ADJP" />
   <xs:enumeration value="ADVP" />
   <xs:enumeration value="PP" />
  </xs:restriction>
 </xs:simpleType>

 <xs:simpleType name="eExcPres">
  <xs:restriction base="xs:string">
   <xs:enumeration value="IIM" />
   <xs:enumeration value="WIN" />
  </xs:restriction>
 </xs:simpleType>

 <xs:complexType name="analysis">
  <xs:complexContent>
   <xs:extension base="act">
    <xs:sequence>
     <xs:element name="type" type="eAnlType" minOccurs="1"
    maxOccurs="1" />
     <xs:element name="subType" type="eAnlSubType" minOccurs
    ="0" maxOccurs="1" />
    </xs:sequence>
   </xs:extension>
 </xs:complexContent>
</xs:complexType>

<xs:simpleType name="eAnlType">
 <xs:restriction base="xs:string">
  <xs:enumeration value="ANOTATE" />
  <xs:enumeration value="TRANSLATE" />
  <xs:enumeration value="CONTEXT" />
  <xs:enumeration value="CLASSIFY" />
  <xs:enumeration value="EXTRACT" />
 </xs:restriction>
</xs:simpleType>

<xs:simpleType name="eAnlSubType">
 <xs:restriction base="xs:string">
  <xs:enumeration value="ADJP" />
  <xs:enumeration value="ADVP" />
  <xs:enumeration value="PP" />
 </xs:restriction>
</xs:simpleType>

<!-- Role -->
<xs:complexType name="document">
 <xs:complexContent>
  <xs:extension base="role">
   <xs:sequence>
    <xs:element name="format" type="eDocFormat" minOccurs="
   1" maxOccurs="1"/>
    <xs:element name="encoding" type="eDocEncoding"
   minOccurs="0" maxOccurs="1"/>
    <xs:element name="uri" type="xs:anyURI" minOccurs="0"
   maxOccurs="1"/>
    <xs:element name="language" type="xs:language"
   minOccurs="0" maxOccurs="1"/>
   </xs:sequence>
  </xs:extension>
 </xs:complexContent>
</xs:complexType>

<xs:simpleType name="eDocFormat">
 <xs:restriction base="xs:string">
  <xs:enumeration value="HTML" />
  <xs:enumeration value="XML" />
  <xs:enumeration value="TXT" />
  <xs:enumeration value="XCES" />
  <xs:enumeration value="GDOC" />
 </xs:restriction>
</xs:simpleType>

<xs:simpleType name="eDocEncoding">
 <xs:restriction base="xs:string">
  <xs:enumeration value="UTF-8" />
```

```
   <xs:enumeration value="UTF-16" />
   <xs:enumeration value="ISO-8859-1" />
   <xs:enumeration value="ISO-8859-2" />
   <xs:enumeration value="ISO-8859-6" />
   <xs:enumeration value="ISO-8859-15" />
   <xs:enumeration value="ASCII" />
   <xs:enumeration value="CP1259" />
  </xs:restriction>
 </xs:simpleType>

 <xs:complexType name="user">
  <xs:complexContent>
   <xs:extension base="role">
    <xs:sequence>
     <xs:element name="id" type="xs:string" minOccurs="0" />
     <xs:element name="username" type="xs:string" minOccurs=
   "0" />
     <xs:element name="userpass" type="xs:string" minOccurs=
   "0" />
     <xs:element name="Email" type="xs:string" minOccurs="0"
    />
     <xs:element name="type" type="xs:string" minOccurs="0"
   />
     <xs:element name="privileges" type="xs:string"
   minOccurs="0" />
    </xs:sequence>
   </xs:extension>
  </xs:complexContent>
 </xs:complexType>

 <xs:complexType name="coordinator">
  <xs:complexContent>
   <xs:extension base="role">
    <xs:sequence>
     <xs:element name="id" type="xs:string" minOccurs="0" />
    </xs:sequence>
   </xs:extension>
  </xs:complexContent>
 </xs:complexType>

 <xs:complexType name="lps">
  <xs:complexContent>
   <xs:extension base="role">
    <xs:sequence>
     <xs:element name="id" type="xs:string" minOccurs="0" />
    </xs:sequence>
   </xs:extension>
  </xs:complexContent>
 </xs:complexType>

 <!-- Entity -->
 <xs:complexType name="endpoint">
 <xs:complexContent>
  <xs:extension base="entity">
   <xs:sequence>
    <xs:element name="address" type="xs:string" minOccurs="0" /> -->
    <xs:element name="protocol" type="xs:string" minOccurs="0" /> -->
    <xs:element name="uri" type="xs:string" minOccurs="0" />
   </xs:sequence>
  </xs:extension>
 </xs:complexContent>
</xs:complexType>

<xs:complexType name="person">
 <xs:complexContent>
  <xs:extension base="entity">
   <xs:sequence>
    <xs:element name="address" type="xs:string" minOccurs="0" />
    <xs:element name="birthDate" type="xs:string" minOccurs="0" />
    <xs:element name="name" type="xs:string" minOccurs="0" />
   </xs:sequence>
  </xs:extension>
 </xs:complexContent>
</xs:complexType>

<xs:complexType name="data">
 <xs:complexContent>
  <xs:extension base="entity">
   <xs:sequence>
    <xs:element name="type" type="eDataType" minOccurs="1" maxOccurs="1"/>
    <xs:element name="encoding" type="eDataEncoding" minOccurs="0" maxOccurs="1"/>
    <xs:element name="data" type="xs:string" minOccurs="0" maxOccurs="1"/>
   </xs:sequence>
  </xs:extension>
 </xs:complexContent>
</xs:complexType>

<xs:simpleType name="eDataEncoding">
 <xs:restriction base="xs:token">
  <xs:enumeration value="base64" />
  <xs:enumeration value="base85" />
  <xs:enumeration value="radix64" />
  <xs:enumeration value="uuencode" />
  <xs:enumeration value="quotedPrintable" />
 </xs:restriction>
 </xs:simpleType>

 <xs:simpleType name="eDataType">
  <xs:restriction base="xs:token">
   <xs:enumeration value="text/plain" />
   <xs:enumeration value="text/xml" />
   <xs:enumeration value="text/html" />
   <xs:enumeration value="text/rtf" />
  </xs:restriction>
 </xs:simpleType>

<!-- === MISC TYPES === -->

 <!--  Message Container  -->
 <xs:element name="iicall-msg">
  <xs:complexType>
   <xs:sequence>
    <xs:element name="act" type="act" maxOccurs="unbounded"
   minOccurs="0" />
    <xs:element name="actRelation" type="actRelation"
   maxOccurs="unbounded" minOccurs="0" />
   </xs:sequence>
  </xs:complexType>
 </xs:element>

</xs:schema>
```

APPENDIX B

The iiCALL Configuration

B.1. iiCALL Configuration (iicall.xml)

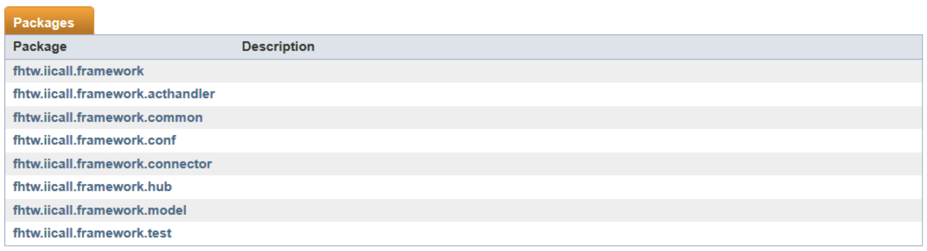
```
<iicall-conf xmlns="http://technikum-wien.at/iicall/framework/conf"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://technikum-wien.at/iicall/framework/conf iiCALLConf.xsd">

 <act-mapping>
   <act-code>ALZ</act-code>
   <act-impl-class>fhtw.iicall.lps.AnalyzeHandler</act-impl-class>
 </act-mapping>

 <connectors>
   <connector-impl-class>fhtw.iicall.framework.connector.HTTPConnector
   </connector-impl-class>

</iicall-conf>
```

B.2. Configuration (iicall.properties)

```
// Connector Configuration HTTPConnetor
fhtw.iicall.framework.connector.HTTPConnector.server.port=8013
fhtw.iicall.framework.connector.HTTPConnector.server.path=/iicall/lps/*
```

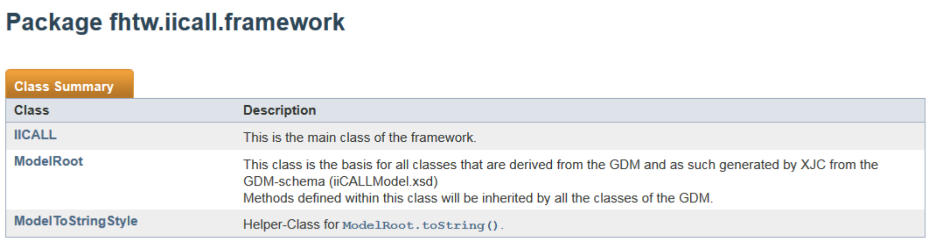
APPENDIX C

The iiCALL Java Packages



Packages

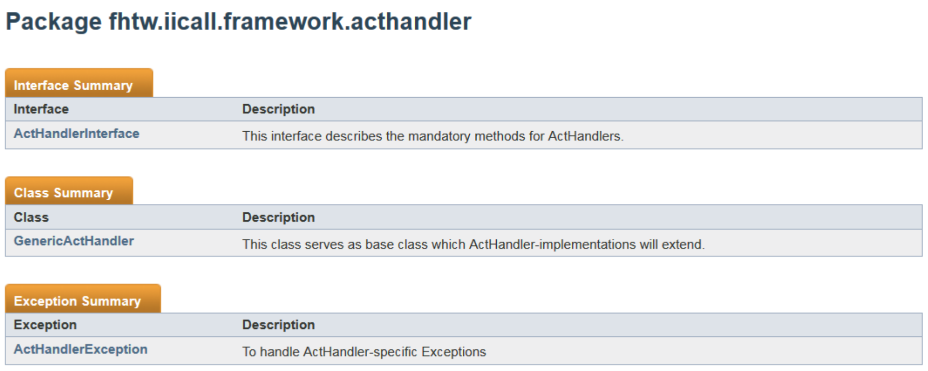
Package	Description
fhtw.iicall.framework	
fhtw.iicall.framework.acthandler	
fhtw.iicall.framework.common	
fhtw.iicall.framework.conf	
fhtw.iicall.framework.connector	
fhtw.iicall.framework.hub	
fhtw.iicall.framework.model	
fhtw.iicall.framework.test	



Package fhtw.iicall.framework

Class Summary

Class	Description
IICALL	This is the main class of the framework.
ModelRoot	This class is the basis for all classes that are derived from the GDM and as such generated by XJC from the GDM-schema (iiCALLModel.xsd) Methods defined within this class will be inherited by all the classes of the GDM.
ModelToStringStyle	Helper-Class for `ModelRoot.toString()`.



Package fhtw.iicall.framework.acthandler

Interface Summary

Interface	Description
ActHandlerInterface	This interface describes the mandatory methods for ActHandlers.

Class Summary

Class	Description
GenericActHandler	This class serves as base class which ActHandler-implementations will extend.

Exception Summary

Exception	Description
ActHandlerException	To handle ActHandler-specific Exceptions

Package fhtw.iicall.framework.common

Class Summary

Class	Description
CorpusUtils	Provides general methods to transform annotated corpus' in various formats.
RecursiveToStringStyle	Helper-Class for `ModelRoot.toString()`.
Serialization	
XMLValidationHandler	This class is used, to log errors and warnings during validation of xml documents.

Package fhtw.iicall.framework.conf

Class Summary

Class	Description
IICALLConf	Java class for iicall-conf element declaration.
IICALLConf.ActMapping	Java class for anonymous complex type.
IICALLConf.Connectors	Java class for anonymous complex type.
ObjectFactory	This object contains factory methods for each Java content interface and Java element interface generated in the fhtw.iicall.framework.conf package.

Package fhtw.iicall.framework.connector

Interface Summary

Interface	Description
ConnectorInterface	This interface describes the mandatory methods for Connectors.

Class Summary

Class	Description
GenericConnector	This class serves as base class which connector-implementations will extend.
HTTPConnector	This class implements a connector for the framework and provides the means to exchange messages over HTTP in XML or JSON format.

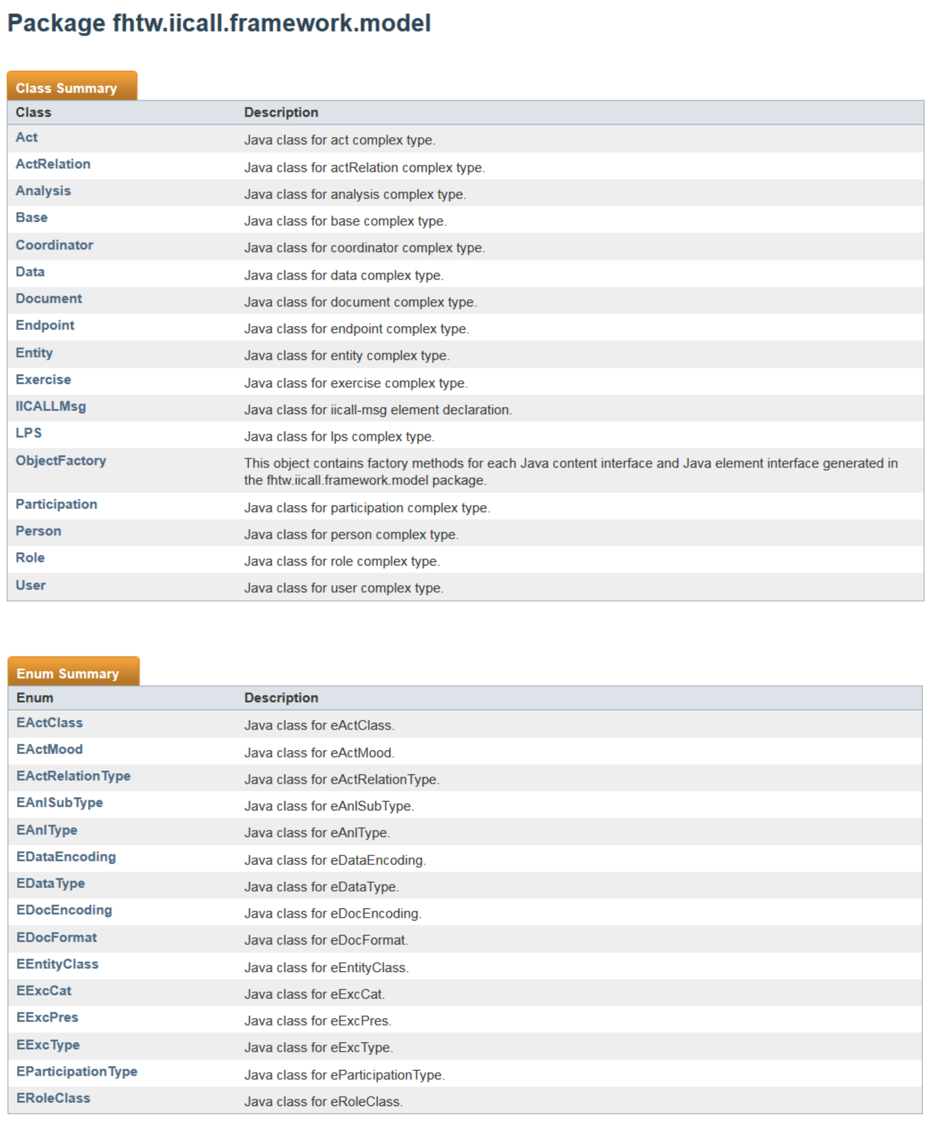
Exception Summary

Exception	Description
ConnectorException	To handle Connector-specific Exceptions

Package fhtw.iicall.framework.hub

Class Summary

Class	Description
Hub	This class serves as hub for iiCALL-messages and serves as bridge between ActHandlers and Connectors.



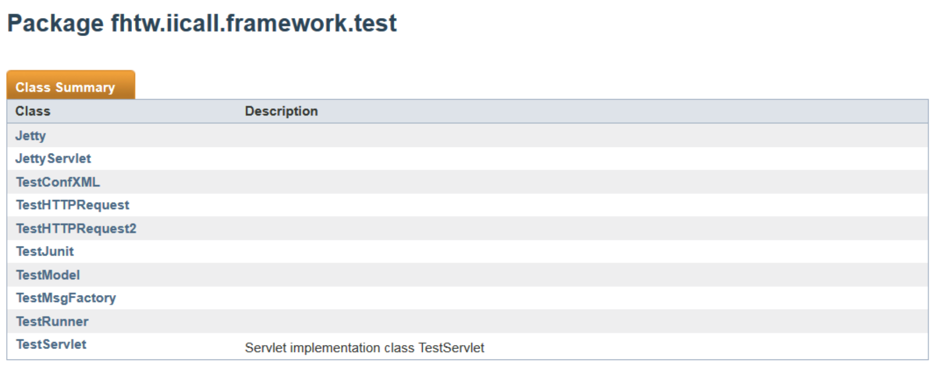
Package fhtw.iicall.framework.model

Class Summary

Class	Description
Act	Java class for act complex type.
ActRelation	Java class for actRelation complex type.
Analysis	Java class for analysis complex type.
Base	Java class for base complex type.
Coordinator	Java class for coordinator complex type.
Data	Java class for data complex type.
Document	Java class for document complex type.
Endpoint	Java class for endpoint complex type.
Entity	Java class for entity complex type.
Exercise	Java class for exercise complex type.
IICALLMsg	Java class for iicall-msg element declaration.
LPS	Java class for lps complex type.
ObjectFactory	This object contains factory methods for each Java content interface and Java element interface generated in the fhtw.iicall.framework.model package.
Participation	Java class for participation complex type.
Person	Java class for person complex type.
Role	Java class for role complex type.
User	Java class for user complex type.

Enum Summary

Enum	Description
EActClass	Java class for eActClass.
EActMood	Java class for eActMood.
EActRelationType	Java class for eActRelationType.
EAnlSubType	Java class for eAnlSubType.
EAnlType	Java class for eAnlType.
EDataEncoding	Java class for eDataEncoding.
EDataType	Java class for eDataType.
EDocEncoding	Java class for eDocEncoding.
EDocFormat	Java class for eDocFormat.
EEntityClass	Java class for eEntityClass.
EExcCat	Java class for eExcCat.
EExcPres	Java class for eExcPres.
EExcType	Java class for eExcType.
EParticipationType	Java class for eParticipationType.
ERoleClass	Java class for eRoleClass.



Package fhtw.iicall.framework.test

Class Summary

Class	Description
Jetty	
JettyServlet	
TestConfXML	
TestHTTPRequest	
TestHTTPRequest2	
TestJunit	
TestModel	
TestMsgFactory	
TestRunner	
TestServlet	Servlet implementation class TestServlet

APPENDIX D

Curriculum Vitae

Personal Data

Name	Dipl.-Ing. Harald Wahl
Date of birth	August, 14th 1969
Place of birth	Linz, Austria
Nationality	Austria
Contact	info@harald-wahl.at

Education

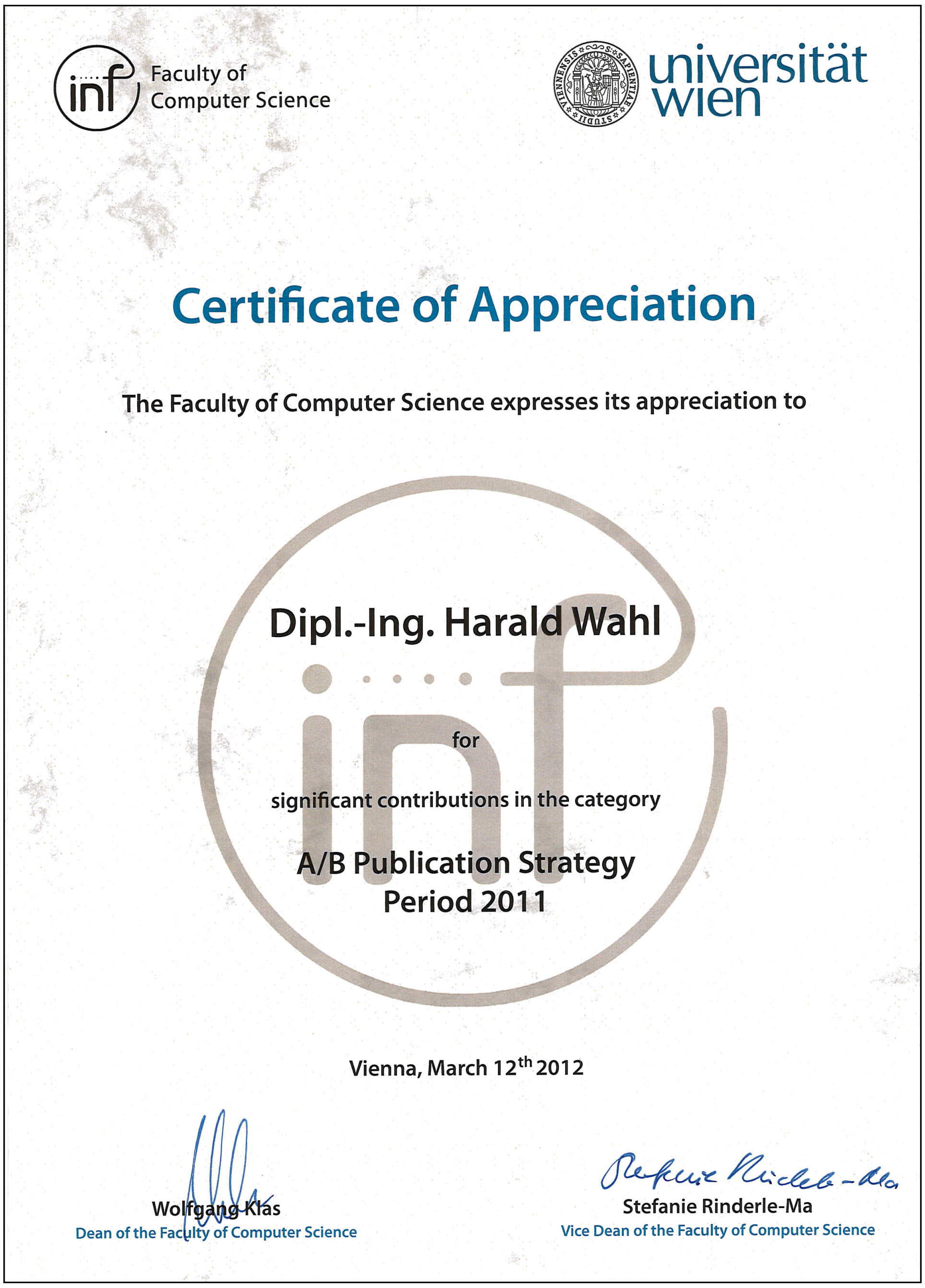
1996	Technical Mathematics at Johannes Kepler University Linz (Information Technologies); Diploma thesis at Siemens, Munich
1993–94	Civil service at Catholic Youth
1987	High school at BRG 2 Linz

Work Experience

Since 2012	Program director at University of Applied Sciences Technikum Wien, Vienna
Since 2010	External lecturer at University of Vienna
Since 2001	Deputy program director at University of Applied Sciences Technikum Wien, Vienna
Since 2000	Lecturer at University of Applied Sciences Technikum Wien, Vienna
1999–2001	Software engineer at Software Competence Center Linz (SCCH), Hagenberg, Upper Austria
1997–1999	Scientific assistant at Institute of Applied Mathematics at Johannes Kepler University Linz, Fuzzy Logic Laboratory Linz (FLLL), Hagenberg, Upper Austria
Freelancer since 1990	Software engineer, Web programmer, graphic designer, ICT trainer and consultant

APPENDIX E

Award



Faculty of Computer Science

universität wien

Certificate of Appreciation

The Faculty of Computer Science expresses its appreciation to

Dipl.-Ing. Harald Wahl

for

significant contributions in the category

A/B Publication Strategy
Period 2011

Vienna, March 12th 2012

[illegible]

Wolfgang Klas
Dean of the Faculty of Computer Science

Stefanie Rinderle-Ma
Vice Dean of the Faculty of Computer Science

Stefanie Rinderle-Ma