Master's Thesis

Configurable User Profiling Service and Simulation of User Behavior in APOSDLE

Daniel Resanovic
dresanov@sbox.tugraz.at

January 25, 2010

Mentor: Dr. Stefanie Lindstaedt
Knowledge Management Institute (http://www.kmi.tugraz.at)
Graz University of Technology

Abstract: In a dynamical world of IT, adaptivity became one of the most important features. To enable adaptivity features of a system there has to be an appropriate infrastructure. Each component should be able to support and contribute to the adaptivity of such systems to a certain extent. How to design a system which is highly abstract and ensures multi purposefulness on the one hand without losing the accuracy of recommendation independent of application domain on the other hand? To overcome this problem CUMULATE (user model component in Knowledge Tree system) introduced the usage of intelligent inference agents responsible for updating a specific property of a user model (e.g. motivation or knowledge level). This thesis introduced a similar approach with the focus not on the property of a user model but on the circumstances under which the user conducts his/her work. One way to achieve this is to have a number of different types of inference agents (user profile component configurations) prepared to work under different circumstances. These circumstances can be either new systems, domains, different states within the same system or domain, working or behavior patterns etc. If for any reason the current configuration is not sufficient we should be able to easily change it with another configuration which is more suitable for the given circumstances. The obvious question at this point would be: how do we know which configurations are best equipped to work under certain circumstances. We have to have some kind of proving mechanism to answer this question. The proving mechanism this thesis presents is a simulation framework specially developed for this purposes. The practical part of this thesis was the implementation of the UPS Prototype 3 together with simulation framework, and the simulation of the user behavior for calibrating the UPS component of the APOSDLE system. Results of conducted simulations showed that we have to consider the so called aging factor in our inference algorithms in order to have the best inferring results. Using this results the number of possible configurations in system was reduced from originally six to just two.

Kurzfassung: In der dynamischen Welt der IT entwickelte sich das Kennzeichen der Adaptivität zu einem der wichtigsten. Um die Adaptivität eines Systems gewährleisten zu können, muss eine entsprechende Infrastruktur sichergestellt sein. Jede Komponente des Systems sollte die Adaptivität zu einem gewissen Grad unterstützen. Wie soll ein System entworfen werden welches einerseits so weit wie möglich abstrahiert ist und ein hohes Maß an Flexibilität bietet, andererseits jedoch die Genauigkeit der Recommendation nicht beeinflusst? Um dieses Problem lösen zu können, haben andere Systeme wie CUMULATE (Benutzermodel Komponente im KnowledgeTree System) so genannte intelligente Inferenz-Agenten vorgestellt. Diese Agenten waren jeweils für eine Eigenschaft des Benutzers zuständig (z.B. Motivation oder Wissen des Benutzers). Die vorliegende Arbeit hat ein ähnliches Konzept verfolgt. Anstatt auf die Eigenschaften des Nutzerprofils wird der Schwerpunkt auf die Umstände/Situationen in denen die Benutzer/Inenn arbeiten gesetzt. Eine Möglichkeit wäre die zu Hilfenahme mehrerer Typen von Inferenz-Agenten (Konfiguration der Benutzerprofil-Komponente), welche für verschiedene Situationen vorkonfiguriert sind. Unterschiedliche Situationen ergeben sich durch neue Systeme, neue Domänen, unterschiedliche Domänenzustände sowie neue Arbeits- und Verhaltensmuster. Sollte die aktuelle Konfiguration aus irgendeinem Grund nicht ausreichend sein, so sollte sie relativ einfach durch eine besser angepasste Konfiguration ausgetauscht werden können. Das Problem dabei ist allerdings, dass nicht bekannt ist, welche Konfiguration für die aktuelle Situation die passendste ist. Es muss demnach ein Überprüfungsmechanismus gefunden werden, welcher sich um diese Problematik kümmert. Dieser Mechanismus wird als Simulation Framework in dieser Masterarbeit vorgestellt. Den praktischen Teil dieser Masterarbeit stellt die Implementation des UPS Prototype 3 und des Simulation Framework dar, und darauf aufbauend die Simulationen von Benutzerverhaltenweisen um die UPS Komponente des APOSDLE-Systems kalibrieren zu können. Die Simulationen zeigen eindeutig, dass jene Algorithmen, welche den sogenannten Aging Faktor berücksichtigen, die besten Ergebnisse erzielen. Mit dieser Erkenntnis wurde die Anzahl der möglichen Konfigurationen im System von ursprünglich sechs auf letztendlich zwei reduziert.

Acknowledgments

I am indebted to a number of people for their support during the research, implementation and writing of my master's thesis.

I would particularly like to thank to:

All APOSDLE team for making a great job on the APOSDLE project.

Special thanks to the User Profiling Service (UPS) team (Günter Beham, Mag. Barbara Kump, DI Christin Seifert) who helped making the UPS Prototype 3 one of the riches (regarding functionalities) and reliable components in the whole APOSDLE system

Günter Beham, the leader of the work package 04 which included UPS, for supporting me throughout the whole process, from design and architecture, over implementation and testing to evaluation of the UPS component. His experience from UPS Prototype 2 helped improving the quality of the UPS regarding functionalities, security, stability and robustness.

Mag. Barbara Kump, for non-technical perspective which enhanced our vision of the UPS. She was also the conceptual creator of the Knowledge Indicating Event approach, which was one of the driving forces for the design of the UPS Prototype 3.

DI Christin Seifert, who came latter into project but provided terrific job in testing and documenting the code. She was also in charged into visualization of the user profile information with so called *Experiences.* This is the client component based on the treemap visualization method.

Dr. Stefanie Lindstaedt, my mentor for the master's thesis and also scientific coordinator for the APOSDLE project, for giving me opportunity to work on this exciting project, and guiding me through my master's thesis.

I hereby certify that the work presented in this thesis is my own and that work performed by others is appropriately cited

Contents

1 Introduction

Due to the increase of information growth as well as the information need, personalization of software became one of the most important topics in recent years. Information has to be tailored/adapted to each individual user. One differentiates between adaptive and non adaptive systems, where adaptive systems are systems which are able to respond to personalized needs of each user. We distinguish between Adaptive Hypermedia Systems, Recommender Systems etc...

It is not just the information growth that causes the need for adaptivity. Computer systems are not easy to learn and once learned it could be easily forgotten. The number of computer systems the individual is exposed to, grows continuously and without adaptivity it would be very hard to keep pace with this development. Even the applications from the same vendor tend to change significantly over time. They are getting new features which need to be learned. There are different levels of adaptivity but all have the same goal, to make it easier for a user to work with such systems.

1.1 Motivation

As mentioned there are different levels of adaptivity. Brusilovsky [10] distinguished two different levels of hypermedia adaptation:

- Adaptive presentation (at content-level)
- Adaptive navigation support (at link-level)

Koch [22] extends these levels with one more, namely adaptation at presentation level. The adaptation of the layout which does not affect the content but rather colors, fonts etc. The first two are more content-based and the third classification relies more on presentation e.g. how to present the information in a way that user finds it very comfortable to work with. Regardless of which level of adaptivity is chosen, there has to be a component which is in charge for the adaptivity capabilities of a system. This component is in many cases based on user model. This makes the user modeling and maintenance of user models some of the most important features in such systems.

User Model tries to provide a model of the user which covers his preferences and needs. Adaptive systems are using this model to make assumptions about the user and to adapt themselves based upon these assumptions [41]. There are numerous application fields for such adaptive systems. Some of them are:

- In E-Commerce personalization is one of the most important tools for customer loyalty. Amazon.com[1] is one example of such personalization. For each purchased item the Amazon gives the recommendation for similar products ("The users which bought this item, bought also the following items as well ..."). Only with user modeling is possible to recommend such user specific products.
- E-Learning is another field where the user personalization is very important factor. In such systems it is crucial to consider previous knowledge of the user (learner) in order to provide the right learning content.

There is a number of questions we have to answer when implementing user modeling based systems. What kind of a system do we want to have? What kind of adaptivity we would like our system to support? What is the domain or scope of our system? How to have a generic system and still preserve the high quality of adaption? How do we get the information relevant for adaptivity? Can we use our system in other domains as well etc?

These are just some of the questions we need to answer before we begin implementing such systems. Even when the system is finished we have many open questions like: are our presumption about user correct, or are our inference tools precise enough etc. Therefore the evaluation of adaptivity is very important in such systems. There are no guidelines which give the answers to all these questions. Thus we have to find out the way to prove the correctness of our system.

1.2 Goals

The main goals of this thesis are to find a new flexible approach in designing and implementing a user profiling component, and a way to prove the correctness of this approach by conducting simulations on the inference mechanism of the user profiling component.

How to design a system which is highly abstract and ensures multi purposefulness on the one hand without losing the accuracy of recommendation independent of application domain on the other hand? One way to achieve this is to have a number of different

[1] Amazon.com:http://www.amazon.com

types of inference agents (user profile component configurations) prepared to work under different circumstances. These circumstances can be either new system, domains, different states within the same system or domain, working or behavior patterns etc. If for any reason the current configuration is not sufficient we should be able to easily change it with another configuration which is more suitable for the given circumstances. The obvious question at this point would be: how do we know which configurations are best equipped to work under certain circumstances? Well, we have to have some kind of proving mechanism to answer this question. The proving mechanism this theses presents is a simulation framework specially developed for this purposes. This framework tries to provide the system with real usage data. The intention is to use simulation of the user behavior to determine which configurations are best suitable for the given circumstances. As a result of such simulations within APOSDLE[2] system (see chapter 7), certain configurations should stand out and be candidates for the final version of APOSDLE.

1.3 Results

The main results of this work are the infrastructure which supports the configurability as a new approach in solving the flexibility issue, and the simulation framework. After applying the configurability approach, we conducted a series of simulations (benchmarking between the different configurations) to narrow down the number of possible configurations in the system. These configurations were the candidates for the final version of the APOSDLE. The results of these simulations showed that we have to consider the knowledge aging factor when implementing the inferring algorithms. Aging factor in this case means the number of user actions used for the inferring purposes. Not just that simulation showed that such algorithms are having better performance but also provided the number of the actions which should be taken into account. At the end we were able to narrow down the number of the possible configurations from original six to just two.

1.4 Structure and relationship between chapters (Organization of the work)

To achieve all defined goals, this work is divided into four major parts which are furthermore divided into chapters. The overall structure with associated chapters is shown in (see Figure 1.1).

[2] APOSDLE system is the adaptive system relevant for this master theses. This system contains user profiling service which is practical part of this theses. http://www.APOSDLE.tugraz.at/

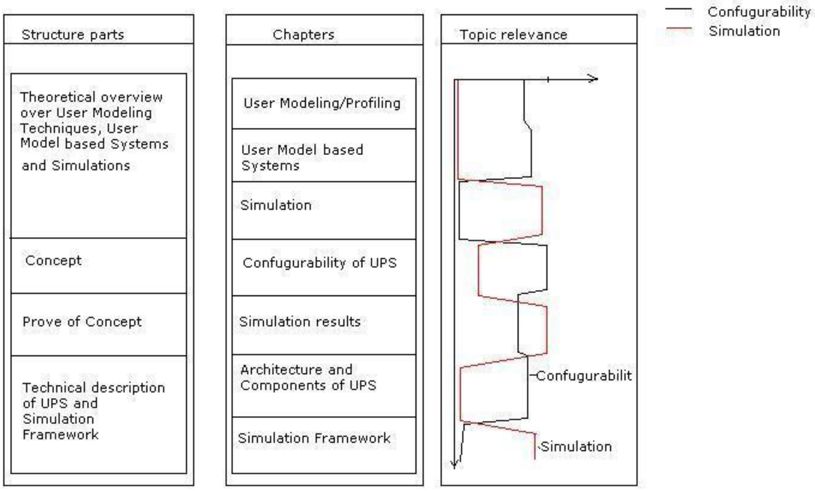


Figure 1.1: Structure of this Master Thesis

The first part is the theory overview and is divided into three chapters. Chapter one is dedicated to an overview over methods and techniques in user modeling. The most popular and used techniques for the user modeling will be explained and partially compared with each other, in order to justify our decision for the user modeling technique used in APOSDLE. The second chapter will give us the overview over the state of the art user model based systems over the past 15 years. The idea is to recognize the developing trends, to identify some of the biggest challenges in such systems and to provide a relation to the APOSDLE system. This chapter can be seen as related work as well. The third and last chapter of the first part is theoretical overview over simulation process. The user simulation is performed as a part of evaluation process (see chapter 6).

The second part is defining a concept of configurable and flexible approach as an answer to the challenges described in the previous chapters. One of the major problems in adapting systems is the incapability to satisfy the accuracy of recommendation on one side and the portability to other domains on the other side. Usually these systems are made for one domain and are not capable of being transported to on another domain (see Chapter 3). Even when made for one domain it is very hard to prove the correctness of inference algorithm[3] in development phase. Usually there has to be some kind of

[3] Note that inference/inferencing algorithm is perhaps not the correct expression. Someone may argue

feedback from the user side, or analysis of an usage data in order to improve or confirm the correctness of present algorithms. At this point it is very expensive to introduce the changes into the system. To reduce these reimplementation/reintegration costs we could have several pre-configured inference agents waiting to be activated if necessary. This concept is described in chapter Configurability of UPS (see chapter 5).

Third part is dedicated to a proof of concept. As mentioned it is hardly possible to confirm correctness of inference process in development phase, therefore we have to wait for the "real data" to be available in order to perform analysis and evaluation. Usually the result of such analysis has to be reintegrated in system. This can be very expensive. With the configurable approach it is possible to have more than one solution/configuration already in system. When the real data are available all it is necessary is to activate the right solution instead of implementing a new from scratch. Simulation of user behavior provides us with such "real data" and can be helpful in reducing the number of possible configurations in system. At the same time it proves the reasonability of this approach by working with different user profile configurations.

The fourth and last part is the technical part describing the general architecture of the APOSDLE system (see chapter 7), design and implementation of the user profiling service in APOSDLE from configurability perspective (see chapter 8) and the simulation framework (see chapter 9). Therein are all relevant technical information starting with which programming language is used, over which framework is used up to some detailed development information.

that it is more heuristic than inference/inferencing but I'm using the inference/inferencing because it is one of the important parts of the inference/inferencing process and therefore the name of the algorithms should somehow incorporate their role in this process.

2 User Modeling/Profiling

A user model is representation of user knowledge and preferences which system believes that user possesses. This information is used to provide adaptivity either by intervention or by co-operative agreement with a user. The information in user model needs to be accurate and actual in order for a system to provide the appropriate adaption [41].

User model consists of static data (long term data) such as birthday, favorite color etc., and dynamic data (short term data), data which are often changed such as user needs, goals, interests etc[19].

Since 1996 adaptive systems are increasing the types of information they are using as basis for their decisions. All these information are saved in user model, (for example goals, knowledge, preferences, user interest, individual character traits)[3, 6]. Over the years more information types are added to user models. The most important are the usage data and usage environment. With these numerous information types it was possible to create more accurate adaptive systems, which ware able to adapt them self to individual needs of users [39].

The knowledge represented in the user model could be acquired either implicitly by inferring about user or explicitly elicited from the user. The explicit knowledge may be acquired with some co-operative behavior such as filling the questionnaire, or from previously stored information about the user. On the other hand the implicit knowledge about the user is acquired by making the inferences about users based on their interactions with the system [42].

One of the major challenges in user modeling is so called Cold Start or Ramp Up problem[12, 23]. The problem is in absence of information about the user in the early phase of user life cycle in the system (initialization phase). This makes the system incapable for adaptation and retrieval of personalized content, and it is specific for recommender systems [12].To overcome this problem, recommender systems have special filtering technique for information filtering [12].

- Explicit, direct user questionnaire (see sub chapter2.6)
- Initial tests of user knowledge and cognition styles [51]

- Stereotype Based Modeling

Main goal of recommender systems is to recommend certain items (books, music, videos, websites etc.) which are from special interest to a user. For example Amazon.com [1] recommends different commodities to purchase based on their interests (books, Cd's, DVDs), YouTube [2] recommends the videos etc.

Recommender systems compare the user profile with different reference characteristics in order to find the best match. There are two different types of filtering.

- Content based filtering [45] compares the characteristics from user profile with the one from items.
- Collaborative filtering [26]compares the user profile with the social environment of the user and tries to find out which items are at most interest for the users from his social surrounding. This could be a group of users with similar interests

There are several approaches/techniques in user modeling. The most prominent are stereotyping(see sub chapter 2.1), feature based modeling(see sub chapter 2.2)(overlay model as special case of feature based modeling) and Bayesian networks (see sub chapter 2.4). The following sub chapters are describing these in detail as well as the difference between terms user model and user profile. At the end of this chapter some acquisition techniques (see sub chapter 2.6) are described.

2.1 Stereotype Based Modeling

Stereotype based modeling is one the oldest techniques used for user modeling [8]. The notation of user stereotypes derives from the work of Elaine Rich[47]. She introduced this approach in GRUNDY system and it is used since in numerous adaptive systems [37]. Stereotype could be defined as structured collection of characteristics or traits, stored as facets mapped to a value, and optionally a confidence-level. Some of the characteristics are triggers which have the potential of firing stereotypes or switching between them. They can be used to "*Provide a way of forming plausible inferences about yet unseen things on the basis of things that have been observed*" [48].

In stereotype based modeling the information acquired during acquisition phase (see sub chapter 2.6) are mapped to adequate stereotype [8]. That means that most appropriate stereotype should be find with best description of the user. The missing information

[1] Amazon.com: http://www.amazon.com

[2] YouTube: http://www.youtube.com

are replaced with the information from the stereotype so long until the new information are acquired (see sub chapter 2.6). Users modeled by stereotype are hierarchically represented in a variety of dimensions and characteristics. At the top of the hierarchy is "(any) person" stereotype which defines characteristics relevant to all users in the hierarchy. The stereotypes lower in the hierarchy may inherit the characteristics from the stereotypes above. The problem may occur if the number of stereotypes from which is inherited is big and some of the inherited characteristics are in conflicted state. There has to be set of pre-defined rules to resolve these conflicts. This is specially the case with the stereotypes placed on the lower level in the hierarchy.

To maintain the adaptivity of the system, user model has to be actual, which means that the changes of the user model have to be continuously evaluated. Based on the trigger this can result in changing of the stereotype assigned to the user.

Although the stereotype based modeling technique for user modeling is one good approach (specially for solving the Cold Start) and it was used in many user model based systems, the feature based modeling technique has recently become dominant technique in user modeling based systems. The next sub chapter describes this technique.

2.2 Feature Based Modeling

As mentioned above (see sub chapter 2.1) feature based modeling has become dominant user modeling technique [8]. Feature based modeling models the user information such as knowledge, interests, goals etc. as so called features. During the interaction with the system these features are continuously changing. For example in E-Learning systems user may forget something (knowledge aging) or can learn some new things. The important thing is to keep the user model actual. Therefore main task of the feature based modeling systems is to monitor user's attributes and to update them if necessary. There are numerous feature types which can be modeled. Some of the most used ones are (a)user knowledge, interests, goals and tasks, (b)background experiences and (c)individual characteristics such as learning style, cognitive factors etc.

User knowledge as attribute tries to model the knowledge of the user over a particular domain. This is especially important by e-learning systems [8]. The knowledge is very dynamic and changeable attribute and has to be continuously monitored and updated. During the interaction with the system, user can easily forget or learn new things. This has to be appropriately represented in user model.

Over the last 10 years *user interests* became very important in adaptive systems. Modeling this aspect of user was not so important in early days of user modeling, instead only learning goals ware modeled. At the moment the importance of user interests modeling surpass, in some cases, the importance of user knowledge modeling [8]. This is primarily the case in systems with the high information density such as encyclopedia, web kiosks, news systems etc.

Goals and tasks as attributes are trying to give a answer on question what the user tries to achieve. These attributes are representing the purpose of user interaction with the adaptive system. Depending on the type of system the goal can be learning goal (e-learning systems), to satisfy information need (search engines) or particular working goal [8, 53].

Background and user experiences attributes model the background information and user experiences which are out of primarily knowledge domain. Therefore it is very similar to user knowledge modeling with the difference (as just stated) that the knowledge which is modeled is not part of primarily knowledge domain [8].

Individual characteristics attribute tries to model the individual user characteristics. Such characteristics are: personal attributes like if the person is extrovert or introvert, which learn style the user has etc. [8].

The individual characteristics, user experience and background are relatively stable information. They change either rarely or not at all. For this reason they belong to statical data. The rest are dynamical data. For APOSDLE system and specially for user profiling service (UPS see chapter 7.2) the dynamical data are of more interest. UPS works almost exclusively with dynamical data to maintain the user profile.

2.3 Overlay Model

Overlay models are very good for Intelligent Tutoring Systems (ITS) [41]. The knowledge of the student (user) is represented as a subset of the knowledge which is overlaid on the expert's knowledge of the domain. The knowledge which is present in expert model and the user do not have it, stands out. The one present in user model is stored as estimation by the overlay model.

In the early day's of overlay model, information contained in the user model, and was related to expert knowledge, were saved in simplified form as yes or no (Boolean). The modern overlay models differentiate between the qualitative (god-average-poor) or quantitative measure (probability that user is familiar with particular concept). The problem with this approach is that the state of the user knowledge is never an exact subset of expert knowledge. User may have different perception of the domain as the experts. There are some attempts to overcome this problem. Perturbation model [33] tries to model this perception which is outside of the expert's view of the domain. To model user misconceptions, Buggy model [13] and Mal rules [31] may be used. The value of overlay models is often questioned and criticized[56, 50] but there are others [4]such as Brusilovsky which thinks that overlay models are powerful and flexible as they measure independently knowledge of the user on different topics.

As we have seen above the most commonly used modeling approaches are stereotype and feature based. Overlay model is derived from feature based model and is of special interest for this thesis because the APOSDLE system use this modeling approach to model the knowledge of the users (see chapter 5).

2.4 Bayesian Networks

In the last decade numerical techniques have become very popular for modeling user's knowledge, goals and identifying the best action to be taken under uncertainty [22]. Bayesian network is one of these approaches.

A Bayesian network is a directed, acyclic graph. The nodes in graph correspond to user properties and the edges to probabilistic influence relationships [1]. These properties may be as well domain knowledge, background-knowledge and/or cognitive model. Each node in graph represents the system's assumption about the possible values of the user properties.

For example if there are two events which could cause a person to be tired, either work or sport. And if the work has direct effect on the fact are we performing sport activity or not (if somebody work he cannot do sport). This situation can be modeled with the adjacent Bayesian network. All three variables have two possible values, true or false.

The names of variables are:

T- Person is tired

S- Sport activities

W-Work activities

Probability function looks like this.

P(T,S,W)=P(T|S,W)P(S|W)P(W)

Otherwise formulated we can say "What is the probability that the person has worked, given the fact that he/she is tired". A simple example is shown in picture (see Figure 2.1)

The next sub chapter tries to explain the difference of the user model and user profile. Although this difference is not of great importance for APOSDLE systems it is necessary to understand that these two terms are not the same.

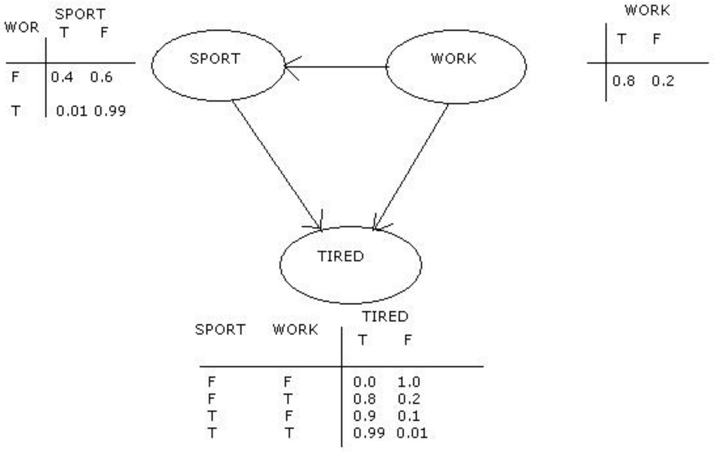


Figure 2.1: Bayesian network

2.5 User Profile

User profile is often used as synonym for the user model. Sometimes they are distinguished to show that user profile is simplified version of user model. Both can be used to represent user's cognitive skills, intellectual abilities, learning styles or preferences etc.

One good way to show the relation between the user model and user profile is to take a look into ITS[3]. There are three types of user knowledge which may be contained in user model (see Figure 2.2)[41].

- Student model
- Psychological model
- User profile

Student models are explicit representations of a state of knowledge of a specific domain and are directly created from the domain model. Domain independent data are either psychological or profile data. Psychological data deals with cognitive traits and are stored in psychological model. There are numerous experimental evidence that the quality of user's interaction with the system depends on their cognitive skills and personality traits [55, 24, 29]. Data concerning interests, background and general knowledge of user is stored into user profile component. Both psychological and profile components needs to be represented explicitly by using some modeling software [41].

[3] Intelligent Tutoring System's

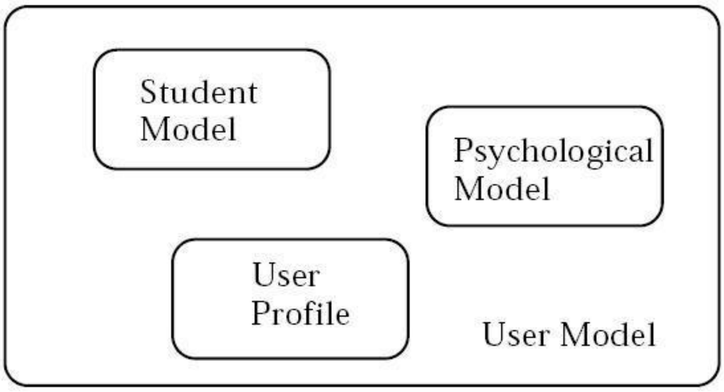


Figure 2.2: Components of a user model [41]

It is important to know that although there is a difference between user model and user profile, in APOSDLE system this difference is of no relevance. These two terms are used as synonyms. The user model in APOSDLE contains student model component and the user profile component or better said the mixture between those two, and it is simply called user profile.

2.6 Acquisition Techniques

There are numerous techniques for acquisition of information about the user. These techniques can be characterized over different dimensions. The following points are describing these different dimensions. Different acquisition techniques are placed within these dimensions. Often comes to overlapping and therefore it is very hard to draw the clear line between them. Different dimensions should be seen as different points of view on different acquisition techniques and not as boundaries for them.

Acquisition techniques can be characterized over several orthogonal dimensions [22]:

1. Active or passive participation of user
2. Direct or indirect acquisition
3. Automatic or user initialized acquisition
4. Explicit or implicit user feedback
5. Logic or probabilistic techniques
6. Online or offline acquisition techniques

2.6.1 Active or passive participation of user

By the active user participation system interacts directly with the user (web interface...). By passive techniques the user information are not acquired through direct user interaction but over inferencing based on user observation. For example the user information can be gathered trough analyzes of HTTP log files, user behavior such as clicks on web sites etc.

2.6.2 Direct or indirect acquisition

By direct acquisition, system collects user data directly from the user feedback. This can happen through questionnaire filled by the user. The questionnaire should be small as possible. If the questionnaire is to long and/or to complex the users may loss their interest for it. Direct acquisition techniques overlap with active user participation.

2.6.3 Automatic or user initialized acquisition

By automatic acquisition, adaptive systems initialize gathering of the user information. User has no influence on this process and has no idea if and when he is observed by the system. Otherwise if the acquisition is initialized from the user and if he can actively participate on building his user profile than we speak about user initialized acquisition. In this case user knows how to influence the user profile. This technique overlaps with the active user participation technique. Both use direct user-system interaction with the difference that in user initialize acquisition technique the user model is changed actively and knowingly, and by direct participation technique user can change its user model indirectly with his/hers behavior change.

2.6.4 Explicit or implicit user feedback

Explicit user feedback means the conscious and intentional information supply by user. On the other hand by implicit user feedback are the information about user gathered based on inconspicuous observation of the user behavior.

2.6.5 Logic or probabilistic techniques

Probabilistic techniques have to be able to deal with uncertainty factor of information in user model. Bayesian networks are able to deal with this type of the user information [17]. On the other hand the overlay model is a good example for a result from logical acquisition technique.

2.6.6 Online or offline acquisition techniques

In online acquisition techniques the user data are directly gathered from active usage of system by the user. An offline acquisition techniques gathers user information from processing of special data bases (for example customer data base).

2.7 Conclusion

One of the first challenges in adaptive systems is the choice of right user modeling technique. Therefore this chapter gave us a short overview over the most popular and used user modeling techniques such as stereotype, feature based and Bayesian networks. Some advantages and disadvantages of these user modeling techniques are presented. Stereotypes are very good in handling initialization of user models so called Cold Start problem. On the other hand the feature based user modeling is very powerful in representing user knowledge in system. It is also the dominant modeling technique in the recent years. The Bayesian networks are very good if we have to deal with probabilistic data. Which one of these modeling techniques to choose is very important and complex issue? The decision is based on the nature of the system and domain in which system has to operate. Sometimes certain forms of hybrid approaches are good solutions (for example stereotyping and feature based) and sometimes just partial aspects of modeling techniques are enough to satisfy the needs of adaptive systems (for example just student model and not all three see figure 2.2). Sometimes is the choice of the proper modeling technique made indirectly by the choice of the acquisition techniques used in system and vice verse. As we have seen (see sub chapter 2.6) above there are different acquisition techniques available. This is the next big challenge in adaptive systems. How well is the user-system interaction or how much does the system wants to molest user with the user feedback? These are the important questions for making the right decision about the correct acquisition techniques.

3 User Model based Systems (State of the Art)

There is a great number of user model based adaptive systems. The user modeling component are more as sub systems to be seen, that are integrated into system and provide basis for the system adaptivity.

This chapter is dedicated in providing an overview over the “trend setters” among user modeling systems over the past 15 years. It is out of scope of this thesis to describe all of the systems present on the market. The idea is to try to identify the development trends in such systems as well as the shared challenges which they all are confronted with. This chapter should also provide an analysis of these systems (to prepare a background) in order to be able to place APOSDLE system among them. The newest systems such as KnowledgeTree or AHA! are described more detailed because of their relevance for APOSDLE system. Others such as UMT, TAGUS or PROTUM are just superficially described in the points which are relevant for the development trends.

3.1 DOPPELGÄNGER (1993)

DOPPELGÄNGER is a User Modeling System design to be able to detect patterns over actions made by users. The main goal of developing this system was to investigate influence of the Machine Learning algorithms on user modeling systems[28]. The system was originally developed to support personalization of daily newspaper.

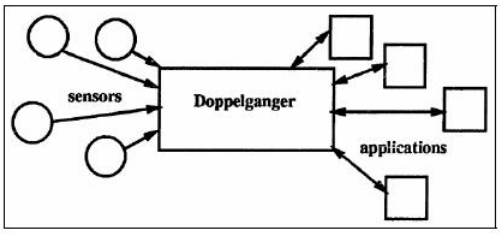


Figure 3.1: The Architecture of the DOPPELGÄNGER[27]

DOPPELGÄNGER's architecture consist of two levels(see figure 3.1). The first one are the sensors which provides the system with the user related information. The second level is the server which collects the information from the sensors, make the inferring and place these inferred information at the disposal of external applications. Sensors can be seen as the specialized unit implementing specific techniques for the information extraction from user activities (e.g. sensor for collecting data about duration of the computer use).

User model developers are using machine learning algorithms in order to generalize sensor data. So made user models were accessible to the users for the inspection and adjustment of their own models[35].

Another interesting feature of this system is the support of data exchange when many DOPPELGÄNGER systems are running simultaneously. The data which are exchanged are the user specific data or the data from a community e.g. teachers, students etc. A user can belong to different communities at the same time. Communities represent the average of their member traits.

3.2 UMT (1994)

UMT[1] is a stereotype based user modeling system. The user characteristics in stereotype are defined as attribute-value pairs. The stereotypes can be formed in any hierarchical order with inheritance capabilities. Each stereotype has a triggering condition, which defines when a certain stereotype should/can be applied to a certain user. Moreover UMT supports rule interpreter which regulates the contradictory inferencing [14].

The assertions made by system are saved/considered as premise or as retractable assumptions (depending on reliableness). Stereotypes which fulfill the activation condition add new assumptions to user model. After each change of user model the inference ruling mechanism infer over premises and assumptions and record the inferential dependencies [34].

Through comparing possible stereotypes with user preferences the appropriate user model will be selected. Presumptions made by UMT have more weight as one coming from stereotype. All calculation steps from all possible stereotypes are saved, which leaves the possibility to re-evaluate them in order to find new user models [34].

[1] UMT-User Modeling Tool

3.3 TAGUS (1994)

TAGUS was primarily developed to support two goals (a) as a framework to represent models of users where meta-cognitive activities of an user were taken into account, and (b) to capture some general mechanisms and techniques for user modeling in the form of services[49].

Basic idea of TAGUS is to achieve some kind of workbench where some techniques of user modeling are implemented and applied. It has a set of services which can be used by persons to test methods or by applications using user models. TAGUS plays a role of user model server.

The architecture of TAGUS is composed of[49]:

- User or Learner model (ULM)
- Maintenance functions
- Acquisition engine
- Reason maintenance system
- Meta-reasoner
- Two interfaces

The assumptions about the user are represented in first-order formulas with meta-operators expressing the assumption types[36]. The system supports the stereotype hierarchy and contains an inference mechanism, a truth maintenance system and diagnostic subsystem. It also supports the user simulations and diagnosis of unexpected user behavior.

3.4 PROTUM (1995)

PROTUM the Prolog based Tool for User Modeling. As stated it is a system based on the Prolog programming language. PROTUM supports the management of stereotype hierarchies (support for multiple inheritances), a system based or predefined rules for resolution of contradictions specific for multiple inheritance and the multiple activation of stereotypes. For each stereotype the systems calculates the activation rate in order to make assumption about the user or to resolve conflicts between inferred stereotypes. In resolving the conflicts PROTUM uses further inferences on the recorded activation rate of stereotypes [34].

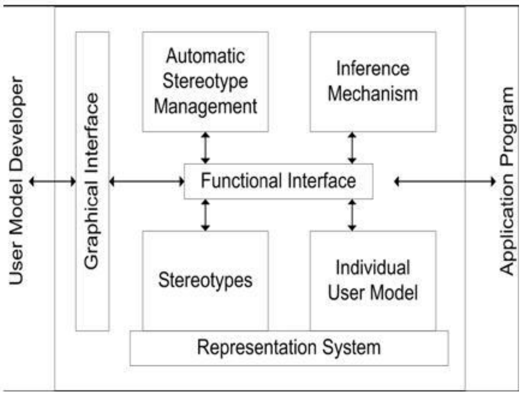


Figure 3.2: BGP-MS User Modeling System [32]

User model in PROTUM is represented as a list of constants where each constant has a type and confidence factor. It is similar to UMT but with better stereotype retraction system [38].

3.5 BGP-MS (1995/1998)

BGP-MS is customizable, application independent system which is able to operate with other systems [38]. It is composed of several units/components with different responsibilities (for example: activation or deactivation of assigned stereotype is one such responsibility). In order to use this system in a specific application domain, appropriate components need to be selected and filled with the domain specific modeling knowledge. BGP-MS includes mechanism for the knowledge representation of its assumption about the user, the domain knowledge of the user and general knowledge of the application domain. In Figure 3.2 is shown how the four core units of BGP-MS can communicate with other applications over Functional Interface.

Representation System is the core unit in charge of creating stereotypes and individual user models. The information in Representation System can be adapted if needed by user model developer through the Graphical Interface.

3.6 ELM-ART(2001)

ELM-ART is an intelligent interactive educational system created to support learning programming in LISP and as an answer to lack of versatility by actual adaptive and intelligent Web-based educational systems (AIWBES). This absence of versatility made them less competitive with course-ware management systems (CMS)[25].

The modern CMS are able to support almost any function of a teacher in the classroom. The teachers could provide a hyperlinked course material, programming examples, quizzes and programming problems to solve. Although completely static it was able to support most of teacher's needs within one single system. Due the advantage of adaptive and intelligent technologies the AIWBES systems were better in performing particular functionalists, but didn't have integrated support for all of them. Therefore versatility was the major driving force in developing the ELM-ART system[25].

ELM-ART provides online learning material in the form of adaptive interactive textbook. The user model is a combination of an overlay model and an episodic student model. ELM-ART uses the user model to provide navigation support, course sequencing, individualized diagnosis of the student solutions and example-based problem-solving support[25].

The first version of ELM-ART was finished in 1996 and was continuously developed and enriched with new features and functionalities over the coming 5 years. One of the first task was to make the system online capable. For that, it was necessary to transform all materials into web compatible form. This was done by dividing the content into small subsections and text pages that are associated with concepts to be learned[25]. These concepts were interlinked to build a conceptual network. The following version of ELM-ART added exercises and tests to the system. By processing the results of these tests the system was able to assess the student's knowledge more carefully. The next version added the multi-layered overlay model. The users were able to declare knowledge units as already known. The system was able to hold not just information that was manually changed by student but also the original state. The students were than able to switch back to original state whenever they wanted without any loss of information[25]. Latest version of the ELM-ART was the basis for the new authoring system NetCoach. With this version the authors were able to create fully adaptive and interactive courses without having to posses any programming skills.

All these upgrades made the ELM-ART intelligent. It can provide several kinds of support usually reserved for the human teachers. The most important features of the system are intelligent problem solving support and intelligent reminder selection. These features are product of a years of research and made ELM-ART unique among other educational systems in the area of programming[25]. Of course there are others intelligent features in this system like capability to build the most relevant learning path for a learner or to determine what the next best learning activity is. The system periodically challenge the learner with a questions that are adapted to the learner's current level of knowledge[25]. The learners are also able to edit their own "learning models" which implicitly changes the navigation structure, sequence of examples and other personalization made by system.

3.7 Knowledge Tree (2004)

KnowledgeTree system is more an architecture for adaptive E-Learning based on distributed intelligent learning activities. The goal is to introduce and use powerful but underused technologies in intelligent tutoring and adaptive hypermedia[7].

Learning management systems (LMS) are dominating on the technology landscape of modern E-Learning. These are powerful integrated systems which supports the variety of activities performed by teachers and student in E-Learning process. LMS offer so-called "one size fits all" service, which means that all learners get the same material regardless of their knowledge, goals or interests. The personalization is not present in such systems [7].

KnowledgeTree is a result of almost 8 years of research on adaptive E-Learning, and efforts against the "one size fits all" approach. It tries to introduce some impressive research results [5] which shows that for every function of LMS there is a number of adaptive web based educational systems (AWBES) which can perform much better. Although these systems exist for over ten years, just a handful of them are actually used for the teaching purposes. The major problem of these systems is not their performance but their architectural constraints. Even though they show impressive results in their function domains, it is very hard if not impossible to integrate them. To cover the needs of Web-enhanced education with AWBES, teacher would need a variety of different AWBES [7].

After exploring several approaches of the component based architecture, the KnowledgeTree was developed. KnowledgeTree is basically a distributed architecture based on re-use of intelligent educational activities [7]. This architecture assumes the usage of at least four types of servers (see figure 3.3):

- Activity server
- Value-adding server
- Learning portals
- Student model

These servers represent the interests of three main stakeholders in E-Learning process: content providers, course providers and learners [7].

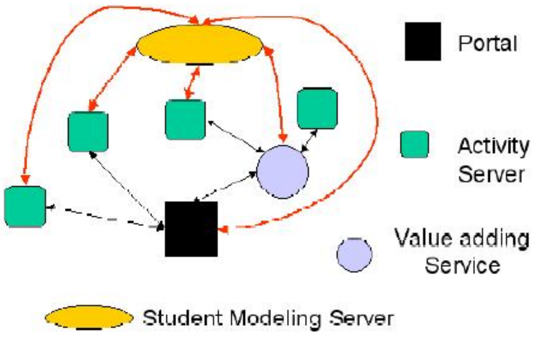


Figure 3.3: Main components of the KnowledgeTree distributed architecture [7]

Learning portal represents the needs of the course providers such as universities or companies. It provides a single-login point for the students to work with different learning tools and content relevant for their curses. It also allows the teachers to access the different content relevant for their courses and to structure them in the way they think is the best for the current leaning purposes. The main difference to the LMS is an architectural separation of the course structure and the reusable content elements [7]. KnowledgeTree uses multiple distributed *activity servers* (services) to provide the learning content and learning support services.

Activity server component play a role similar to an repository for the reusable leaning content and service providers. The difference to the traditional learning repository is that unlike traditional repositories, which are basically pools for storing simple and static learning objects, an activity server host highly interactive and adaptive learning content. It is also possible to host interactive learning services such as forums or shared annotations. The duty of an activity server is to answer to the requests made by learning portals or value-adding service for a specific activity. The concept of reusable activities should encourage the content providers to produce advanced, interactive learning content and services. So delivered content and service activities can be intelligent and highly adaptive. In particular, each activity can obtain information about the student from student model server and provide a highly personalized learning experience. Activity server also monitors the progress of the student, changes of the student goals, his knowledge and interests, and then sends updates to the student model server[7].

Value-adding service combines the features of a portal and an activity server. It process the content and services adding some valuable functionality to it such as annotation, visualization or content integration. These services are course-neutral and therefore can

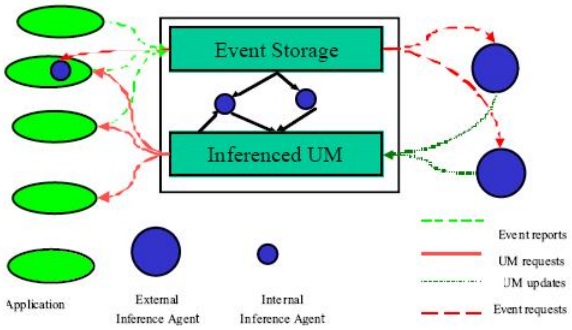


Figure 3.4: Centralized student modeling in CUMULATE [7]

be re-used in multiple courses[7].

The *student model server* represents the needs of the student in the E-Learning process and allows distributed E-Learning to be highly personalized. It is maintained by a university (provider) or by student themselves. The *student model server* collects data from portals and activity servers and provides information about the student to the different adaptive portals and activity servers. They use this information for adapting the learning content and services to the needs of the each individual student. It is very important to have this component centralized for the whole system so that each learning activity can get access to all student related information i.e. student progress. To fulfill these requirements KnowledgeTree has its own *student model server* called CUMULATE.

3.7.1 CUMULATE

The CUMULATE is a event-based centralized student modeling architecture [7]. The idea of this server is to collect events about student learning activities from different servers and to process them into shareable set of student parameters(see figure 3.4).

Each event is specified by the activity server (i.e., page is read, question is answered). Event is time-stamped and stored into event storage part which is normally implemented as relational database. The flow of events is than processed by different internal and external inference agents which than update the values in the inference model part of the server. The updated information is either in the form of property-values pairs or property-object-value triples. Each inference agent is responsible for updating a specific property. One agent can for example process the event sequence trying to deduce the current motivation level while a other inference agent may infer the knowledge level of the student for each topic in the course. These inferencing agents do not have to reside exclusively on the student model server side but also on the external servers within the system. The current version of CUMULATE uses two different inference agents[7].

A knowledge-inferring agent process events made by user a related to a topic. This agent extracts information from different learning activities and tries to deduce the current student knowledge level of this topic[7].

A activity-inferring agent process learning activity related events. It counts the number of visits and annotation for this activity and calculates activity levels[7].

Additionally CUMULATE provides a number of maintenance tools for administrators as well tools for teachers and students to examine the content of student model[7].

3.8 AHA! System(2000/2007)

AHA![2] is an open source general-purpose system and was developed to support an on-line course with some user guidance. Meanwhile the system has become numerous extensions and tools that made AHA! into versatile adaptive hypermedia platform[20]. The development started in year 2000 and continuously enhanced over the coming 7 years. The latest version of AHA!3.0[3] was finished in July 2007.

AHA! is based on overlay user model2.3 and exploits a model of user traits such as goals, preferences and learning style for adaptive purposes. Domain model consist of concepts and their relationships[21]. Each fragmented information presented in webpage is related to the corresponding concept. The concept definition includes name, resource (reference to the corresponding information (page) in information space) and owns some predefined attributes[40]:

- Access - as an indicator to start the adaptation engine when a certain concept is accessed
- Suitability - express if the page assigned to the concept is suitable for presentation
- Knowledge - an integer value which corresponds to the user's knowledge level of particular concept
- Visited - the information which indicates if the user already visited resource related to the concept

Beside these predefined attributes a user can define his own attributes[40].

[2] The Adaptive Hypermedia Architecture

[3] http://aha.win.tue.nl/

Concepts are connected between each other through concept relationships defined an form of adaptation rules such as (a) prerequisite, (b)knowledge propagation and (c) knowledge update[40].

The adaptation model consists out of events and condition action rules and is responsible for adaptation of the knowledge represented in the adaptation model. The condition is expressed as a Boolean value using concept's attributes and the action related to this event. The definition of the attribute in the domain model contains also event-condition-action rules which define how the values of the attributes in user model should be updated[40]. When certain event occurs (i.e. page is shown) rules associated with this attribute are activated and if the conditions related to this rule are fulfilled, a certain actions are triggered. Action defines which value and how this value should be changed resulting with a possible change of knowledge level of the user.

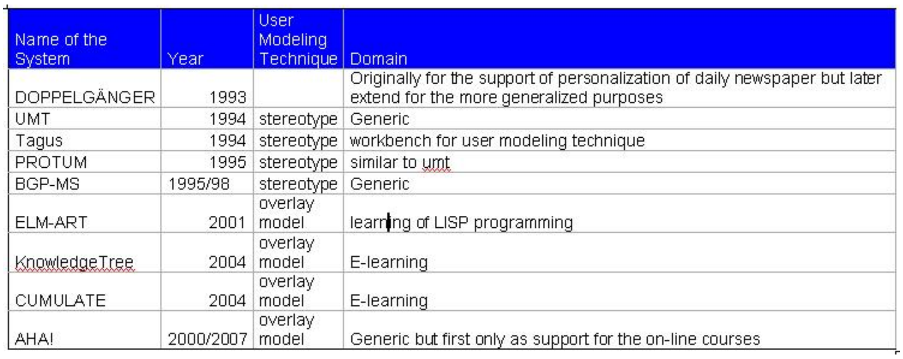
Depending of the knowledge level of the user certain information-fragments are (not) shown. E.g. for beginners the information fragments such as hints should be showed, on the other hand for student with expert skills this information may be redundant[40]. Estimation of the knowledge level is a complex task. It is done by making the student perform various on-line tests. There is also a feedback mechanism which enables the student to inform the system whether she/he understands presented information. This will help the inferring mechanism to make better assumption about student's knowledge level.

3.9 Conclusion

The previous sub chapters showed us that there are numerous user model-based systems on the market. Each of them has its own approach in trying to improve their adaptivity. There are some general challenges which they have to confront with like choosing the appropriate user modeling technique, trying to make their architecture component-based or to try to make the system more generic. Some of the trends in development are easy to recognize the other not. For example it is clear that trend in user modeling techniques moved from stereotype based onto overlay model(see figure 3.5).

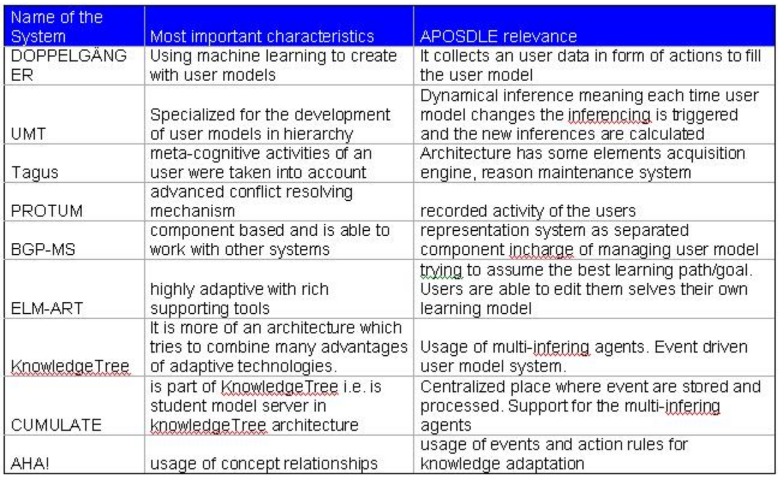
Since begin of the last decade the overlay model was used exclusively. This is one of the reasons why this technique has been chosen for the APOSDLE system. The table (see figure 3.5) shows us that most of the systems are limited to certain domain but also that most of the systems which are longer in use having trend toward generalization of appliance field (see sub chapters 3.8 and 3.5).

On the other hand there are new creative approaches presented in the newest sys-



Name of the System	Year	User Modeling Technique	Domain
DOPPELGÄNGER	1993		Originally for the support of personalization of daily newspaper but later extend for the more generalized purposes
UMT	1994	stereotype	Generic
Tagus	1994	stereotype	workbench for user modeling technique
PROTUM	1995	stereotype	similar to umt
BGP-MS	1995/98	stereotype	Generic
ELM-ART	2001	overlay model	learning of LISP programming
KnowledgeTree	2004	overlay model	E-learning
CUMULATE	2004	overlay model	E-learning
AHA!	2000/2007	overlay model	Generic but first only as support for the on-line courses

Figure 3.5: Comparison of User Model-based Systems



Name of the System	Most important characteristics	APOSDLE relevance
DOPPELGÄNGER	Using machine learning to create with user models	It collects an user data in form of actions to fill the user model
UMT	Specialized for the development of user models in hierarchy	Dynamical inference meaning each time user model changes the inferencing is triggered and the new inferences are calculated
Tagus	meta-cognitive activities of an user were taken into account	Architecture has some elements acquisition engine, reason maintenance system
PROTUM	advanced conflict resolving mechanism	recorded activity of the users
BGP-MS	component based and is able to work with other systems	representation system as separated component incharge of managing user model
ELM-ART	highly adaptive with rich supporting tools	trying to assume the best learning path/goal. Users are able to edit them selves their own learning model
KnowledgeTree	It is more of an architecture which tries to combine many advantages of adaptive technologies.	Usage of multi-infering agents. Event driven user model system.
CUMULATE	is part of KnowledgeTree i.e. is student model server in knowledgeTree architecture	Centralized place where event are stored and processed. Support for the multi-infering agents
AHA!	usage of concept relationships	usage of events and action rules for knowledge adaptation

Figure 3.6: Relevance of different systems to APOSDLE

tems which are really completely new approaches in solving adaptivity issues in user model-based systems. It is obvious that the two newest systems KnowledgeTree (with CUMULATE as student model server) and AHA! (see figure 3.6) using the event-driven approach, meaning that there are certain user actions which are defined as events and the inferring mechanisms are processing these events trying to infer knowledge level of the user. KnowledgeTree (and CUMULATE) is using a multiple inferring agents approach. The developers of this system recognized that there is a need for more general approach in inferring process. Therefore they introduced the possibility to have more inferencing agents specialized for certain tasks. There is no limitation for the number of possible inference agents. At the moment they are using two in CUMULATE system. This is also of special interest to the APOSDLE system where the similar concept was used to support the flexibility of the system.

4 Simulation

In general we can say that simulation is the imitation of real things, processes or state of affairs where the purpose of simulation is the representation of certain key characteristics or behaviors of selected physical or abstract system over time[2]. Roger D. Smith gave the definition of simulation as "*Simulation is the process of designing a model of a real or imagined system and conducting experiments with that model. The purpose of simulation experiments is to understand the behavior of the system or evaluate strategies for the operation of the system...*"[52]. Mathematical algorithms and relationships are made to describe assumptions made about the system. Thus the basis for a model is made which can reveal how the system works. If the system is simple we can represent and solve this model analytically but in most cases the problems of interest are much more complex so that a simple mathematical model cannot represent them. In such cases the system behavior can be estimated by means of simulation[52].

Simulation can be used in different contexts such as modeling of natural or human systems in order to better understand their functioning. There are also numerous other contexts such as technology simulation for performance optimization, testing, education and training. Basically the simulations are used to show the possible real effects of alternative conditions and actions.

The following sub chapter 4.1 explains the process of simulation. The most important steps in designing and running one simulation are explained. The sub chapter 4.2 provides an insight into most relevant simulation languages and tool-kits available for the simulations.

4.1 The Simulation Process

In the last several decades the process of creation and operation of a simulation has suffered sever changes. In the beginning of simulation days this process was reserved only for the experienced practitioners. Now days this process has evolved and there are clearly defined steps for developing, validating, operating and analyzing the results of simulations (see figure 4.1). These steps are:

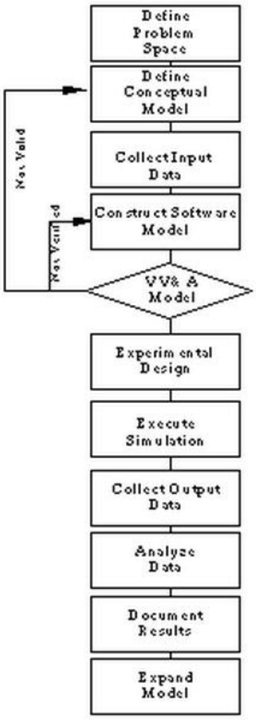


Figure 4.1: Simulation Development Process [52]

- Define/Formulate the Problem Space
- Setting of objectives and overall project plan
- Model conceptualization
- Data collection
- Construct Software Model (Model translation)
- Verify, Validate and Accredit the Model
- Design Experiments
- Execute Simulation
- Collect Output Data
- Analyze Data
- Documentation and Reporting
- Expand Model

4.1.1 Define/Formulate the Problem Space

First, the problem that needs to be modeled has to be defined. Also the goals and requirements must be stated along with the desired accuracy of the results. The clear boundaries have to be defined between the problem we are addressing and the surrounding environment. In order to interact with external systems (beyond the boundaries of the system) there should be clearly defined interfaces in the system[52].

4.1.2 Setting of objectives and overall project plan

Setting of objectives means identifying the questions which should be answered by the simulation study. The project plan should include the various scenarios that should be in investigated. It also includes the time, personnel, hardware and software requirements, stages in the investigation, output at each stage and costs the simulation will bring[2].

4.1.3 Model conceptualization

After defining the problem one or more appropriate conceptual models can be defined. Conceptual models are used to abstract the system under investigation. Such models include the algorithms to be used for the simulation, required input data, and the generated output. This phase is also used for documentation of the assumptions made about the system along with potential effects on the simulation result caused by these assumptions. Each limitation caused by model, data or assumptions must be clearly defined so that the simulation can be properly determined[52][57].

It is recommended to begin simply and that the model should easily grow until the appropriate complexity of the model has been reached[2].

Each conceptual model defined must include a description of the amount of time, number of persons involved and equipment required to produce and operate the model. Then all models need to be compared in order to find the single best solution which meets the goals and requirements of the problem[57].

4.1.4 Data collection

After determining the solution space, the data needed to operate and define the model have to be collected. There is several data types required[52]:

- Information as input parameters
- Information as aid for development of algorithms
- Information which is used to evaluate the performance of the simulation runs
- Known behaviors of working systems
- Information on the statistical distributions of the random variates to be used

This phase is the most perceptible to error and misapplication, while collecting the correct input data is one of the most difficult (sub)processes in the simulation process.

4.1.5 Construct Software Model (Model translation)

Based on the solution defined and data collected, the simulation model is created. The creation of the computer simulation follows the same process as developing of any other software product[52].

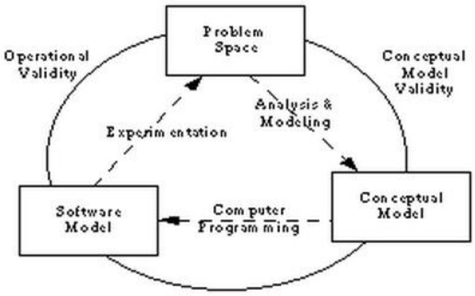


Figure 4.2: Model Verification[52]

4.1.6 Verify, Validate and Accredit the Model

One of the essential phases of simulation process is the phase where the problems for which models where created are verified, validated and accredited. This process ensures that the model algorithms, input data and design assumptions are correct and are able to solve the problem defined at the beginning of the process[52][18].

The simulation development process is divided into the problem space, conceptual model and software model with defined relations between these phases in order to make verification, validation and accreditation phase easier to conduct (see figure 4.2).

Validation is the process which should determine if the conceptual model correctly addresses all issues stated in the problem space. It is also used to determine if the software model consistently and correctly reflects the real world. This is usually done by comparison and experimentation with a know data set.

Verification is the process where it determined if the software model accurately represents the conceptual model.

Accreditation is an official process of accepting the software model as representative model for the specified purposes.

4.1.7 Design Experiments

This phase is used to identify the most productive, accurate and cost acceptable (if the simulation runs are expensive) methods for the running of the simulation. To reduce the number of runs there are statistical techniques which can be used[43]. For each simulation scenario there are decisions to be made. They are concerning the length of simulation run, the number of runs and the manner of initialization[2].

4.1.8 Execute Simulation

This is the phase where the simulation is actually executed. The simulation runs generate the output data needed to answer the problems addressed by the model. In cases of certain model such as Monte Carlo,[1] a many hundreds or even thousands of runs may be needed to achieve statistically relevant results[52][43].

4.1.9 Collect Output Data

Collection of output data is done parallel to the execution of the simulation. The output data is collected, organized and stored for the further processing.

4.1.10 Analyze Data

Collection of simulation output data is a process that is distributed over time. Therefore analyzes must be performed to recognize the long term trends and to provide answers to the questions which motivated the construction of the simulation. This process can produce information in different forms for displaying purposes (tabular, graphics, map, animation etc...)[52].

4.1.11 Documentation and Reporting

It is very important to document the results of the simulation study. The results have to be divided to interested parties, which needs to identify the degree to which the simulation was able to give answers to specific questions for the possible improvements[52]. Documentation is also very important if the simulation model is going to be reused by other parties, or if the model is going to be modified. The reporting should be clear and concise[2].

4.1.12 Expand Model

Creation of simulation models is a very expensive and difficult process. Therefore we need to reuse the present models on related projects if possible. The whole process will be redone and adopted to new requirements[43].

These steps are some general guidelines to be used in creation of the simulations. The following sub chapter describes some of the tools and languages used in/for simulations.

[1] The Beginning of the Monte Carlo Method (http://library.lanl.gov/cgi-bin/getfile?00326866.pdf)

4.2 Simulation Tool-kits and Languages

There are number of different simulation languages and packages/tool kits developed for simulation purposes. These languages are developed to serve specific domain problems, rather than support general purpose programming. The general programming languages can be, and are widely used for programming simulations but in domains for which simulation specific languages or packages do not yet exists or in cases where there is no economical gain for developing one (for example by very specific and unique problems)[52].

The most popular simulation languages are[46]:

1. *Simula* was the first simulation programming language and was developed by O.J.Dahl from the Norwegian Computer Center in 1967. It is more of general purpose programming language with some extensions to support simulations. It was one of the first object-oriented languages and was used as motivation for the later developed C++ language.

2. *GPSS/H* was developed at IBM in 1969. This language supports interactive debugging, various trigonometric and statistical functions. It also automatically collects the simulation output data[54].

3. *SIMSCRIPT II.5* is an event and process-oriented programming language. It was developed at the Rand Corporation in 1962. SIMSCRIPT can be used to support different types of simulation such as discrete-event, continuous and combination simulations.

4. *SIMAN/Cinema* was developed by Systems Modeling and is a combination of simulation and animation language. The models are design with the graphical interface and then automatically converted into code.

5. *SLAM II* made by Pritsker Associates and is used for process-oriented simulations. With some extensions can support also event-oriented simulation and combination of these two. The models are represented in network form with nodes and branches, which can be drawn by the developer and then converted into code.

6. *MODSIM* from CACI is an object-oriented programming language with powerful graphic extensions, compiler and debugger.

7. *ACSL* (Advanced Continuous Simulation Language) was developed to support continuous simulations. It is used for modeling time-dependent, nonlinear equations and transfer functions.

8. *CSMP* (Continuous System Modeling Program) is built on basis of the three general types of statements: (a) *structural* to define a model, (b) *data* to assign the numerical values to parameters and (c) *control* to manage model execution.

Due the complex syntax the usage of these languages was not easy. They are in many cases more complex then general purpose programming languages. This fact served as motivation for developing more easy to use graphic packages and tool-kits. In time several of those were developed[52][15]:

1. *That* interactive, visual simulation package for both discreet and continuous modeling which allows users to build models using graphical interface design for this purpose. It can be extended and fine tuned by adding C or FORTRAN routines.

2. *Workbench* is simulation environment which allows building of complex models using graphical visual interface. The models are specified as a hierarchy of directed graphs.

3. *TAYLOR II* is a graphic model building package using four fundamental entities (elements, jobs, routines and products). These entities are manufacturing oriented.

4. *COMNET III* was made by CACI Products and was design to simulate communication networks.

5. *BONeS Designer* is used to model the protocol and messaging layers of computer architectures and communication systems.

6. *CSIM18* is a library used to describe different activities and statistical distribution of microprocessors, communications, transportation's and manufacturing systems.

7. *SimPack* was developed by University of Florida intending to support the development various simulation programs.

8. *CPSim* provides an execution kernel that manages scheduling, deadlock prevention and message passing.

9. *VRLink* is toolkit for interactive simulations. It is mainly used for distributed military simulations.

10. *ITEMS* is also toolkit for interactive simulations and is used for construction of virtual worlds and entities to populate them.

11. *MultiGenII* is used a 3D modeling tool for the visual representation of simulated objects.

There are many different languages and tool-kits used for simulation. To describe them all in detail is not in scope of this thesis. The intention of this sub chapter is to attempt to provide some insight into current situation in the simulation field.

5 Configurability of UPS

As stated in chapters two and three before implementing an User Model-based System we have to confront several challenges. Some of them are more general nature and the other ones are more domain specific. Regardless of type, we have to be able to deal with them. This chapter describes the concept of a flexible approach in dealing with some of these challenges. This flexible approach concept (configurability) brings not just answers to the challenges just mentioned but also some open questions. To answer these questions we need to have a mechanism to evaluate this approach.

The first sub chapter is divided into two parts. The first part summarize all challenges in developing of user model-based systems, and tries to argument the solutions implemented in APOSDLE, where the second part tries to answer the challenges related to the configurability approach itself. KIE Approach sub chapter explains the concept of using the user's actions as events which indicates knowledge of the user in the given context. This concept is the basis for the configurability of UPS in APOSDLE. Sub chapters 5.3 and 5.4 are describing the two main factors in this configurability approach, namely chain models and inferencing algorithms. Chain models are the representation of the event-knowledge level mappings, and the algorithms are the core of the inferencing intelligent agents. The last sub chapter presents some of the possible configuration examples to better understand the underlying concept.

5.1 Challenges

Configurability as a new approach in UPS is presented as an answer to challenges related to the general as well the APOSDLE specific issues. This approach also brings some open questions itself. To make it easier to analyze and answer all these challenges/questions it is necessary to divide them on two levels. The first level deals with the more general issues such as challenges related to user modeling techniques or to user model based systems. The second level tries to summarize the challenges related to configurability approach.

5.1.1 APOSDLE System challenges

Based on the challenges related to the general as well APOSDLE specific issues, there are four questions to be answered:

- Which user modeling technique should be used?
- How do we collect data about the user (acquisition technique)?
- What is the scope of our system (domain specific vs. generic)?
- What are the newest trends in the state-of-the-art systems?

These questions can be seen as four dimensions of a four dimensional space where each dimension is somehow related and dependent to other three. For example, in order to make the best decision on which user modeling technique to use, we should know the scope of our systems, which acquisition techniques are used and the newest trends in this area. For example the chapter about user modeling based systems (see chapter 3) provide an overview over the various state of the art user modeling systems, where we can identify several trends. First it is obvious that the overlay model technique became dominant over the past several years. All systems build over past eight years are using overlay model for the user modeling purposes (see sub chapter 3). It is also clear that most of the systems are made for a specific domain and are not able to support more generic approach. Some of the systems which are further developed are trying to surpass this obstacle and to become generic. There is a trend to expand the scope of the user modeling systems, from single domain to multi domain capable systems (see 3.8, 3.5 and 3.7).

APOSDLE answers to these questions/dimensions are:

User modeling technique: The user model in APOSDLE should represent the user knowledge in the domain. As stated in chapter about user modeling techniques (see chapter 2) the feature based modeling and specially the overlay model as a special case of the feature based modeling, is the best suitable for this. Also as stated above there is a clear trend in using this technique in the newest state of the art systems. Therefore APOSDLE uses overlay model as representation of the user knowledge throughout the system.

Acquisition technique: Considering the fact that some of the domains in APOSDLE could have several hundreds of different topics, it is very unpractical to conduct explicit information acquisition. If questionnaire is used to test the user knowledge over the domain, he/she would be probably forced to invest a lot of the time in answering them.

This way the usability of the system would be seriously damaged. Therefore the APOSDLE tries to collect the usage data implicitly and to reduce explicit data acquisition as possible. By doing so, user can focus himself to exploit many features APOSDLE offers. This is very important for the configurability approach while so implicitly collected data (see KIE sub chapter 5.2) are representing the basis for the Chain Model definition and therefore the backbone of the configurability.

Scope of the system: This is perhaps the most important issue and one of the greatest challenges in implementing an user modeling system (see chapter 3). APOSDLE tries to overcome the single domain problem and to become a generic system able to support many different domains. The scope of the APOSDLE system was always multi domain, meaning as long there are working processes present in a company, APOSDLE should be able to support the learning process during the work. The same nature of APOSDLE is therefore generic. All APOSDLE needs in order to work over different domains is the underlying model of the company. The description of this modeling process is out of scope for this thesis and therefore will not be elaborated. For more details see APOSDLE Deliverable on modeling tools[1].

In prototype 3 there is an attempt to extend this generic approach to user profiling service. To provide generic capabilities of UPS we followed a so called configurability approach. One configuration consists (a) of a mapping of each KIE to a certain knowledge level (in the system known as Chain Model) and (b) one inference algorithm.

There are two driving forces for the configurability approach. First as just mentioned APOSDLE is a generic system and we have to provide a more generic approach for user profiling. This means that differences between the domains could be reflected on the process of inferring. Even in the same domain, depending on the usage of the domain (behavior pattern), it could be needed to have more flexible inferring mechanism in the background. A mechanism able to detect this difference and apply the best suitable inferring algorithm. Second, it was not possible to know what kind of the inferring mechanism is the best for APOSDLE without having some kind of benchmarking between different inferring mechanisms. To do so we also need a real data to feed our inferring algorithms. This second aspect was the dominant force in implementing the configurable approach for the user profiling service (UPS) in APOSDLE P3. This means that the UPS could be configured from outside and not directly from code (hard coded). All we have to know are the configuration parameters (chain model plus inference algorithm) and we can switch between them as we find suitable. The challenges related to configurability approach are described in sub chapter 5.1.2.

[1] APOSDLE Deliverable: http://www.aposdle.tugraz.at/media/multimedia/files/third_prototype_aposdle

Trends: Some of the trends are all ready mentioned in this thesis, but there are some new creative and unique approaches in the newest state of the art user modeling systems. For example KnowledgeTree (see sub chapter 3.7) defines an user actions as so called events to feed his inferring mechanism. The inferring mechanisms are called inferring agents and they can be different and work together to provide the best result. These features could also to be found in AHA! (see sub chapter 3.8) system. This is important while the APOSDLE systems uses similar concept for his UPS. The events as basic units for inferring algorithm are something new in APOSDLE P3. More about these events and their impact in the KIE sub chapter (see sub chapter 5.2).

5.1.2 Challenges for Configurability

Configurability offers some answers to the previously stated challenges. Specially if considering the scope issue of the system. But this approach raises some question itself:

- If there is a possibility to have multiple configurations in the system, where and how to manage them?
- How to change between these configurations if needed?
- Which one to take if there are many?
- How to import new configurations if necessary?

Answering these questions requires new technical and conceptual solutions in APOSDLE P3 (Prototype 3).

If there is a possibility to have multiple configurations in the system, where and how to manage them? This is a technical question. To hold and manage multiple configurations in the system we need to have flexible underlying infrastructure. Specially the data model needs to be flexible and extendable (see sub chapter 8.2). One configuration consists out of the chain model (see sub chapter 5.3) and the algorithm (see sub chapter 5.4). Chain models are preserved in a data base and can be easily extended if necessary, where the algorithms are programmatically implemented. The information about which configuration to use, meaning which combination of chain model and algorithm to use is present in external configuration file. The application reads this configuration file in run time and then activates the correct configuration.

How to change between these configurations if needed? As just mentioned there is a configuration file which holds the information which configuration should be used. To use different configuration all what is necessary is to change this configuration file to points to other configuration. This is one of the major advantages of this approach. It is not necessary to change things programmatically. If the present configuration does not work properly, it is possible to change the used configuration in configuration file in run time. Meaning the application does not needs to be recompiled or stopped at all.

Which one to take if there are many? This is the most important and difficult question. To answer this question there has to be some kind of benchmarking present in the system. To conduct benchmarking (comparison) between different configurations two things are needed.

First, the defined and mapped KIE to knowledge level as a part of Chain Model (see sub chapters 5.2 and 5.3) and implemented algorithms (see sub chapter 5.4).

Second, the (real) data are necessary. The simulation framework is implemented to provide the system with such data. Only after comparing different configurations it is possible to say which ones are the candidates for the final version of the APOSDLE.

How to import new configurations if necessary? As mentioned in the first question/answer it is requirement to have flexible and extendable underlying infrastructure. To import new configuration means to import new chain model and algorithm. If these are not already in the system, we need to import new chain models in data base and implement new algorithm programmatically. Importing new chain models in data base is a simple process of updating the data base. The implementation of the new algorithm requires more effort. If the algorithm is implemented all we need is to add him to other algorithms already implemented without having to change anything else in system. This is possible due the usage of so called strategy design pattern[2].

5.2 KIE (Knowledge Indicating Event) Approach

One of the most important issues in the user modeling systems is to keep the user profile up-to-date. In order to do so APOSDLE collects data either implicitly by observing the interactions of the user with the system, or explicitly by requesting feedback directly from the user. APOSDLE tries to reduce the explicit acquisition and puts the focus on implicit data acquisition. This is primarily done through observation of the user activities. These activities are introduced as KIE (knowledge indicating events). The number and their part in APOSDLE system changed or better said evolved during the development phase. Originally there were 21 KIE which UPS used to infer the knowledge level of the user. Meantime some of the KIE changed their status to different type of events, so called logging events. These are all those events which are important for the

[2] design pattern: http://www.javaworld.com/columns/jw-java-design-patterns-index.html

better understanding of user-system interaction but are very difficult to map to certain knowledge level. This is a needed condition for one event to be KIE.

The conceptual creator of the KIE approach is Mag. Barbara Kump. This concept is similar to concept of events used in KnowledgeTree and his student model server CUMULATE (see sub chapters 3.7 and 3.7.1). Using this concept enables the flexible in scalable approach in implementing user profiling services. Flexible while it is possible to change or vary the event-knowledge level mapping and therefore try out different mappings, and scalable while the number of KIE-s determine the complexity or preciseness of the inferring process in the background.

KIE are the basis for the one of two factors important for the configurability of the UPS in APOSDLE. The so called chain models used in APOSDLE are different mappings between the KIE or the combination (sequence) of KIE and their knowledge level they are indicating. The next sub chapter is dedicated to explain the part of the chain models and their mappings for the configurability of UPS.

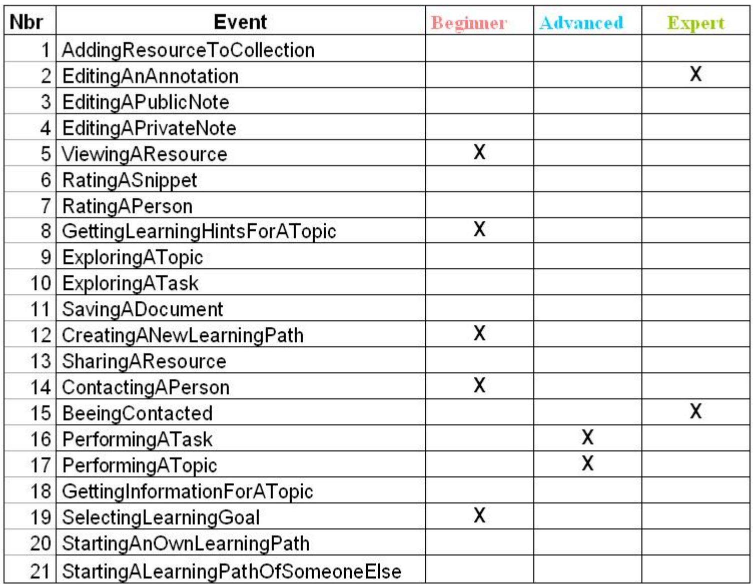
5.3 Chain Models (Mappings)

As mentioned in the previous sub chapter, chain models are built from KIE or the sequential combination of those. This means that we can map directly each KIE to a particular knowledge level for example *EditingAnAnnotaiton -> Expert*[3] or to map a certain sequence of KIE to a knowledge level for example *PerformingATask-SelectingLG-ViewingAResource -> Advanced*. The idea behind sequencing the events into a chain (therefore chain model as name), is to implement certain system usage constraints direct into the chain model and to try to reduce the “noise” (dirty data) used for the inferencing purposes. This means that user needs to use the system in a predefined way (path) in order for his/hers activities to be taken into account by the inferencing mechanism. Unfortunately this second approach was not exploited enough because of the lack of resources and time to test these chains in a “real world” situation(see sub chapter 6). For the configurability approach only single event chain were used. Nevertheless the different mappings were used in order to test and prove this concept (see sub chapter 5.5).

There are basically two problems in defining a chain model.

- Is it possible at all to map a KIE (user activity) to a knowledge level? There are no heuristics which proves that certain user action reflects the knowledge of the user. And if it is possible to map an action to a knowledge level then we have to answer the following question.

[3] EditingAnAnnotation is a example of the KIE. The list of KIE is shown in figure 5.1



Nbr	Event	Beginner	Advanced	Expert
1	AddingResourceToCollection			
2	EditingAnAnnotation			X
3	EditingAPublicNote			
4	EditingAPrivateNote			
5	ViewingAResource	X		
6	RatingASnippet			
7	RatingAPerson			
8	GettingLearningHintsForATopic	X		
9	ExploringATopic			
10	ExploringATask			
11	SavingADocument			
12	CreatingANewLearningPath	X		
13	SharingAResource			
14	ContactingAPerson	X		
15	BeeingContacted			X
16	PerformingATask		X	
17	PerformingATopic		X	
18	GettingInformationForATopic			
19	SelectingLearningGoal	X		
20	StartingAnOwnLearningPath			
21	StartingALearningPathOfSomeoneElse			

Figure 5.1: The first proposal for the events

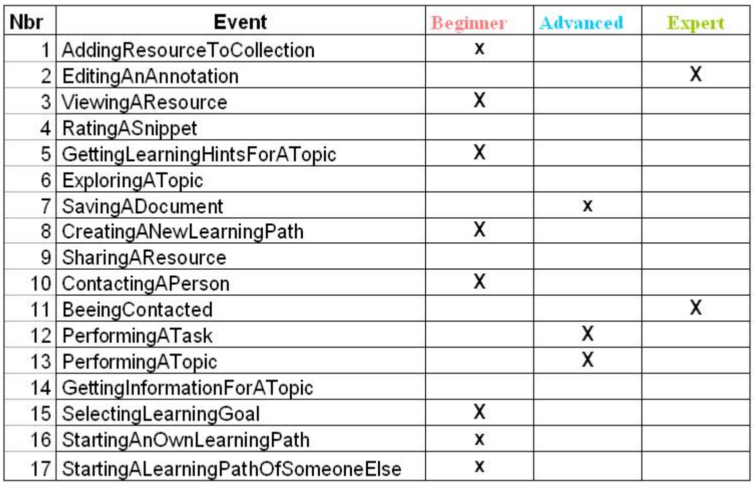
- To which level to map? The issue is similar as above. We can argue that certain activities in a particular domain meaning a certain level of knowledge but there are ho hard evidences to supports this hypothesis.

Exactly here, the first advantages of using configurability approach are getting obvious. If we have several possible mappings we can try out each of them, and then based on the result we become and the result we are expecting, determine the best possible mappings. This is a simplified version of configurability approach just to show how this approach can be helpful.

The first chain model had 21 events (see Figure 5.1). It was obvious that some of the KIE are not easy to map to a particular level and for some events was not even sure if they are in fact the KIE. Therefore the first chain model was never actively used in system but for testing purposes.

After analyzing the first KIE set used for the first chain model it became obvious that some of the KIE in the list do not exist in the system. Due the technical restrictions it was not possible to detect these events at all. Therefore the list was reduced to 17 events (see Figure 5.2). This process of changing the list of the KIE is a process which evolved over time. After working with the system our knowledge and understanding of the system became better which eventually result in changing the number of KIE as well as the mappings to particular knowledge level.

As a consequence of this evolving process the final version of the chain models was



Nbr	Event	Beginner	Advanced	Expert
1	AddingResourceToCollection	x		
2	EditingAnAnnotation			X
3	ViewingAResource	X		
4	RatingASnippet			
5	GettingLearningHintsForATopic	X		
6	ExploringATopic			
7	SavingADocument		x	
8	CreatingANewLearningPath	X		
9	SharingAResource			
10	ContactingAPerson	X		
11	BeeingContacted			X
12	PerformingATask		X	
13	PerformingATopic		X	
14	GettingInformationForATopic			
15	SelectingLearningGoal	X		
16	StartingAnOwnLearningPath	x		
17	StartingALearningPathOfSomeoneElse	x		

Figure 5.2: The evolution of event list

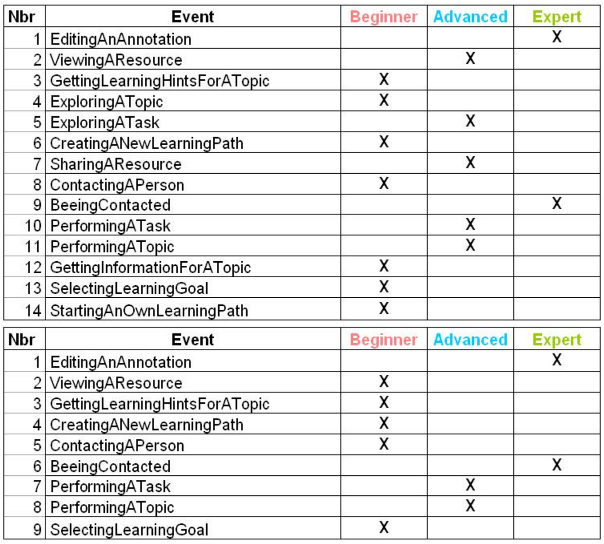
reduced to 14 and 9 KIE respectively (see Figure 5.3). These two chain models differentiate from each other in number of the KIE and mappings to knowledge level as well, and are candidates for the final chain model used in APOSDLE system. The decision which of these two should be used was made after conducting simulations (see chapter 6). The next sub chapter explains the second variable parameter relevant for the configurability concept, namely inferring algorithms.

5.4 Algorithms

The inferring algorithms are not only the second (variable parameter) but also the most important factor in configurability concept. Chain models are the underlying data structures upon the inferring intelligent algorithms are working, where the inferring algorithms are the heart of so called inferring agents.

Basically there are no limitations of how many algorithms can reside in the system. The problem is not just in implementing these inferring algorithms but also in validating and testing them (see chapter 6).

Similar to CUMULATE (see sub chapter 3.7.1) the inference algorithms in APOSDLE are based on different approaches. Some like FREQUENCY are simple ad hoc math where other like EVENT_WEIGHTING having Bayesian Networks[11] as an inspiring idea[9]. Important to say is that due the configurability approach we have the freedom of trying out different types of algorithms like machine learning or algorithms used in information retrieval (WEIGHTING).



Nbr	Event	Beginner	Advanced	Expert
1	EditingAnAnnotation			X
2	ViewingAResource		X	
3	GettingLearningHintsForATopic	X		
4	ExploringATopic	X		
5	ExploringATask		X	
6	CreatingANewLearningPath	X		
7	SharingAResource		X	
8	ContactingAPerson	X		
9	BeeingContacted			X
10	PerformingATask		X	
11	PerformingATopic		X	
12	GettingInformationForATopic	X		
13	SelectingLearningGoal	X		
14	StartingAnOwnLearningPath	X		

Nbr	Event	Beginner	Advanced	Expert
1	EditingAnAnnotation			X
2	ViewingAResource	X		
3	GettingLearningHintsForATopic	X		
4	CreatingANewLearningPath	X		
5	ContactingAPerson	X		
6	BeeingContacted			X
7	PerformingATask		X	
8	PerformingATopic		X	
9	SelectingLearningGoal	X		

Figure 5.3: Event-Mappings used for simulation

There are six inferring algorithms implemented in APOSDLE. They are different in logic and type. Logically there are only three different algorithms and they are: FREQUENCY, WEIGHTING, EVENT_WEIGHTING. Each of them has two types: DEFAULT meaning that all recorded events are taken into account by inferencing mechanism, and RECENT_NUMBER_OF_EXECUTIONS meaning that only a predefined number of the recorded events are taken into account by inferencing mechanism. Combining these two elements we get the following six inferring algorithms:

- FREQUENCY
- WEIGHTING
- EVENT_WEIGHTING
- FREQUENCY_RECENT_NUMBER_OF_EXECUTIONS
- WEIGHTING_RECENT_NUMBER_OF_EXECUTIONS
- EVENT_WEIGHTING_RECENT_NUMBER_OF_EXECUTIONS

Although the UPS was a part of Prototype 2 (P2) the UPS in prototype 3 (P3) was completely new developed. The reason is that UPS P3 was based on the different concepts. The concept used as basis for the P3 UPS was new, namely KIE approach, which made the inferring algorithm in P2 obsolete. Nevertheless I tried to import the basic idea of P2 inferring algorithm into P3 UPS. This was the first inferring algorithm in P3.

5.4.1 FREQUENCY

This algorithm is the legacy algorithm from P2 and was the first implemented algorithm in P3. Basic idea is to count the user activities in a particular topic based on their knowledge level mapping. For example, if the user performs 10 activities in a particular topic, the system converts these activities into KIE and the inferring algorithm counts how many of these events are mapped to BEGINNER, ADVANCED or EXPERT knowledge level respectively. The knowledge level assigned to the user for the topic is the level in which the number of events was greatest. This algorithm was relatively simple and easy to implement. The problem was the fact that the number of KIE-s mapped to knowledge levels was quite different. For example the chain model which uses 14 KIE-s in total had following distribution over knowledge level: BEGINNER-7, ADVANCED-5 and EXPERT-2. It seemed obvious that the BEGINNER-s events are having more chance to be executed by the user than the other ones. This was the motivation to introduce the second algorithm in system.

5.4.2 WEIGHTING

WEIGHTING algorithm tries to balance these unfavorable relations between knowledge levels. Therefore each event belonging to different knowledge level type was weighted with the knowledge level type weight. For example if there are 7 events from totally 14 belonging to BEGINNER then each event from this type will be weighted with 1/7. Respectfully for the ADVANCED the weight attached to these events would be 1/5, and for EXPERT 1/2. This way the balance was made and statistically each KIE in system had the equal impact on the inferring process. The idea for this algorithm came from the *Maximum tf Normalization* algorithm used in information retrieval. *Maximum tf Normalization* normalize the tf[4] weights of all terms occurring in a document by the maximum tf in that document [16].

5.4.3 EVENT_WEIGHTING

This algorithm was a result of a deeper analyzes of the system usage. Although some KIE are apparently having more chance be executed then the others based on the knowledge level type, it is not quite so. For example PerformingATask or PerformingATopic are

[4] tf: Term Frequency

members of ADVANCED knowledge level type and if used in WEIGHTING algorithm are less weighted then some member of the BEGINNER knowledge level type. But in the fact these two KIE are having far more better chance to be executed than for example GettingLearningHints (from BEGINNER type) because the user is confronted with these two activities right on the beginning/front of the APOSDLE interface. These two actions are actually the entry points into system (if using APOSDLE Suggest see sub chapter 7), meaning if the user wishes to perform any other activity he/she has to perform one of these two activities. This makes it obvious that the approach used for the WEIGHTING algorithm was perhaps not the optimal one. The idea in EVENT_WEIGHTING algorithm is to pursuit the weighting concept but on the more granular level, on the KIE level. The obvious question at this point is how we can determine which weights individual KIE-s should have. There are no heuristics supporting this. Therefore it was necessary to analyze the usage of the system.

This algorithm was inspired by Bayesian Network algorithm [30] (see sub chapter 2.4). The basic difference is that EVENT_WEIGHTING algorithm does not use probabilities but the weights for labeling the nodes. Therefore the formula used for calculating the weights is different than the one for calculating probability by Bayesian network. But the idea behind is similar. The weight of a node (KIE) is depending on the weight of following nodes (in Bayesian Network depends on the predecessor Node).

The first step was to identify the usage paths present in the APOSDLE system (see Figure 5.4). Secondly the weights for the actions based on the usage paths trough APOSDLE system needed to be derived.

To do so the graph approach was used. The paths are represented as a branches of a graph (see Figure 5.5). For example if the user would like to Edit a Annotation (over APOSDLE Suggests) he/she has to: Perform A Task -> Select LG -> View a Resource -> Edit Annotation. Since Edit a Annotation KIE is present in fourth level of the graph the user has to go through previous three levels to reach this event. This means although the user intention was to annotate a certain part of a document he/she is forced to activate other events along the path in order to do so. This increases the weight of these events. Obviously there is direct relation between the weight and the level in which certain event is placed. I introduced the following formula in order to reflect this dependency.

Ni = (Numb. of level - (i-1)) + Sum(Outgoing Nodes)

For example for the VR - Viewing a Resource:

RT = (4 - (4-1)) = 1

EA = 1

GLH =1

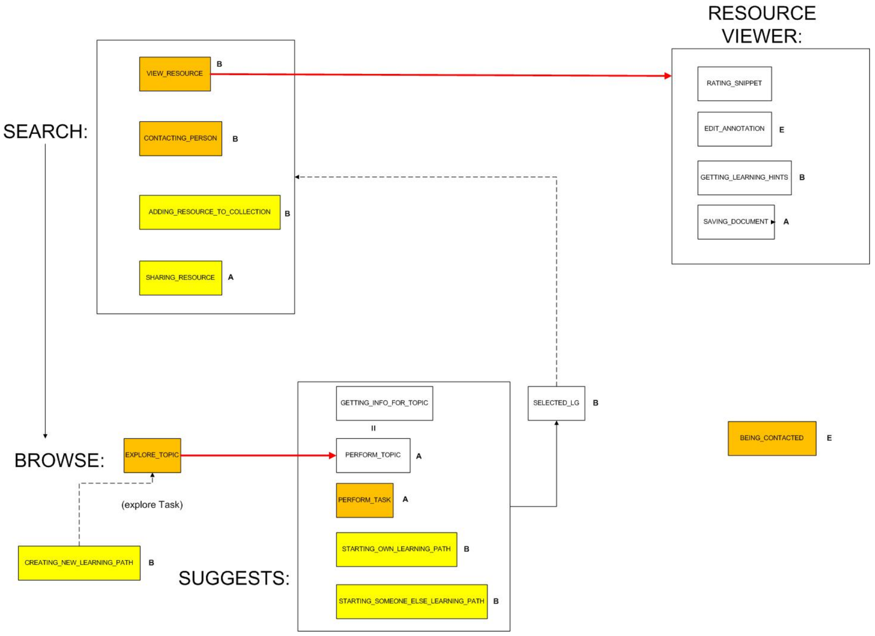


Figure 5.4: APOSDLE System Usage Paths

SD = 1

VR = (4 - (3-1)) + RT + EA + GLH + SD

VR = 2 + 1+ 1 + 1 = 5

As we see the weight of leafs elements in fourth level is always 1. Therefore the weight for the VR event is the sum of his level weight (in this case 2) and the weights of all outgoing nodes 3x1=3. This was a simple example just to show how the weights are calculated. The whole process of calculating weights is not finished. There is also a graph for the SEARCH usage path. This graph is not so complex but has impact on the whole weighting. After calculating the weights for the SUGGEST usage we have to add the weights from the SEARCH using the same formula. The complete formula looks like this.

Complete Weight for KIE = 1/(Weight for Suggest for KIE + Weight for SEARCH for KIE)

5.4.4 RECENT_NUMBER_OF_EXECUTIONS Algorithms

The last three algorithms are basically the same as these previously three with just one difference. The previously described algorithms (FREQUENCY, WEIGHTING, EVENT_WEIGHTING) are using the whole recorded history of user actions for the in-

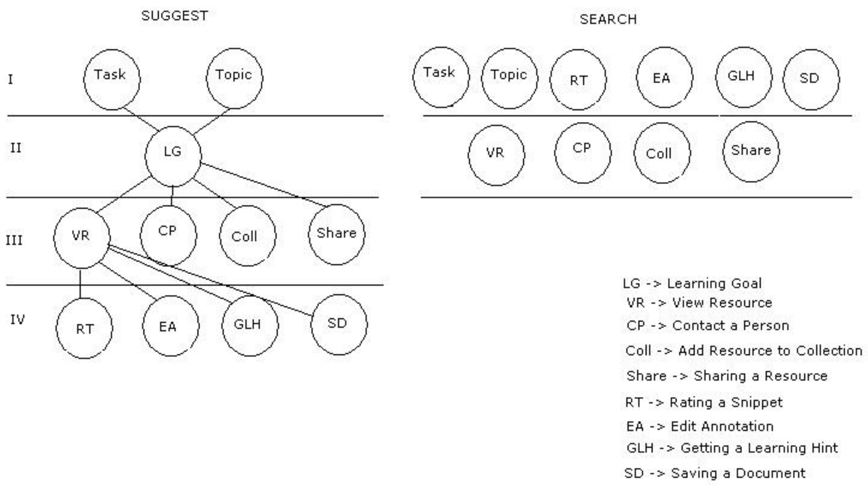


Figure 5.5: Usage Paths through APOSDLE as graphs

ferring purposes where the other three (FREQUENCY_RECENT_NUMBER_OF_EXECUTIONS, WEIGHTING_RECENT_NUMBER_OF_EXECUTIONS, EVENT_WEIGHTING_RECENT_NUMB
are using predefined number of these actions.

Expert systems which maintain the knowledge about certain topics across the time period, must have an awareness of time. This awareness can be made by incorporating time factor or so called knowledge aging in our algorithms[44].

The obvious problem was how many actions in the past should be taken into account. If too many actions are taken into account the inferring process will be burdened by the information which is not relevant any more. On the other hand if too few are taken than valuable information are lost and the inferring process is not precise. We used the simulations to find out the right answer to this question (see chapter 6). The simulations showed that the optimal number of the actions to be taken into inferring process is between 40 and 50. This result is then used to initialize these algorithms. And in fact the algorithms of this type showed the best performance and are used as the final candidates for the APOSDLE.

5.5 Examples of Configuration

For better understanding of the configurability concept this sub chapter will try to describe couple of examples of how the chain models can be combined with the inferring algorithms and what conclusion can be drown out of it. It is important to state that the number and the nature of chain models and inferring algorithms are not restricted by any means. This way we can really cover all individual needs of all possible domains. APOSDLE system has two chain models and six inferring algorithms. That means that there are twelve possible configurations in system. Each of this configuration has it owns advantages and disadvantages. The two of them are explained in order to get the "feeling" about the approach.

Configuration one:

14-Events Chain Model and the EVENT_WEIGHTING_RECENT_NUMBER_OF_EXECUTIONS algorithm. This combination brings a very fine tuned approach. Firstly the 14-Events chain model is the model with more events than the other chain model (just 9 events) meaning better sensitivity and more data for the inferring mechanism. Secondly the inferring algorithm is designed for really fine granular inferring purposes on the event level. This algorithm is also good for the learning purposes due to inferring over the restricted number of events made in the past. The impact of to old events is hereby reduced and just the newer actions are taken into account. This is simplified version of dealing with the knowledge aging problem. The downside of this combination is that the falsely weighted or mapped events are having more impact on the result. The mapping and weighting process is extremely important. The inferring process in extreme cases could have impact on the system performance.

Configuration two:

9-Events Chain Model and the FREQUENCY algorithm. This combination is very simple. The number of events used is small and the algorithm is primitive. Obvious advantage is that user can easily understand underlying process and get more in touch with the system. The downside is the disadvantage by determining the fine tuned differences. It is perhaps too robust for the actual usage. Impact on the system performance is minimal due the simplicity of inferring process.

Working and benchmarking more of these combinations is described in following chapter. The following chapter is about simulation experiment conducted for the evaluation purposes of UPS.

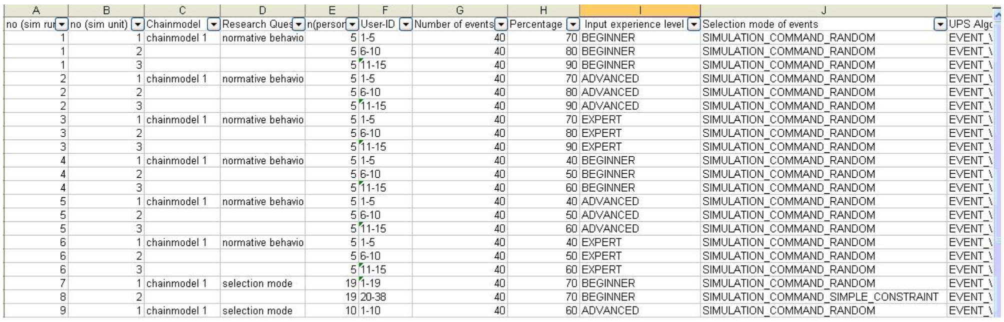
6 Simulation results

The previous chapter describes the underlying concept of this thesis, namely configurability of UPS. One configuration consists (a) of a mapping of each KIE to a certain knowledge level (in the system known as Chain Model) and (b) one inference algorithm. Since there are multiple inference algorithms and chain models the goal is to find out which of these combinations are best suited for the APOSDLE system. To do so "real" usage data is needed which can be used to benchmark different configurations. Because we wanted to calibrate the UPS before real users were using it, we introduced a simulation framework with the goal to create/simulate the real usage data. Data should always be simulated for one knowledge level (BEGINNER, ADVANCED or EXPERT). Normative behavior was varied, that is, the extent to which a person showed KIEs that were related to his or her knowledge level. For instance, if the behavior of an EXPERT was simulated and 90% out of the simulated user actions (KIE) were "EXPERT actions", we speak about a simulation of an EXPERT with 90% normative behavior.

There are three simulation modes in simulation framework representing the degree of system restrictions in creating data. They are:

- Random mode, meaning that events are generated randomly without any system constraints at all.
- Simple constraint mode, meaning that some of system constraints are implemented in process of creating events. This mode covers most of APOSDLE Search constraints (see sub chapter 7). Certain events cannot be executed without some preconditioning events. For example user can not edit an annotation without opening a document or snippet before.
- Complex constraint mode, meaning that APOSDLE system constraints are fully considered. This mode covers entire APOSDLE Suggest constraints. Due the complexity of these constraints it was not possible to implement them programmatically. They have to be created manually.

The basic idea of increasing the degree of constraints is to see how important certain system constraints are and what kind of impact does this have on simulation results.



A	B	C	D	E	F	G	H	I	J	
no (sim ru	no (sim unit)	Chainmodel	Research Ques	n(person	User-ID	Number of events	Percentage	Input experience level	Selection mode of events	UPS Algo
1	1	chainmodel 1	normative behavio	5	1-5	40	70	BEGINNER	SIMULATION_COMMAND_RANDOM	EVENT_\
1	2			5	6-10	40	80	BEGINNER	SIMULATION_COMMAND_RANDOM	EVENT_\
1	3			5	11-15	40	90	BEGINNER	SIMULATION_COMMAND_RANDOM	EVENT_\
2	1	chainmodel 1	normative behavio	5	1-5	40	70	ADVANCED	SIMULATION_COMMAND_RANDOM	EVENT_\
2	2			5	6-10	40	80	ADVANCED	SIMULATION_COMMAND_RANDOM	EVENT_\
2	3			5	11-15	40	90	ADVANCED	SIMULATION_COMMAND_RANDOM	EVENT_\
3	1	chainmodel 1	normative behavio	5	1-5	40	70	EXPERT	SIMULATION_COMMAND_RANDOM	EVENT_\
3	2			5	6-10	40	80	EXPERT	SIMULATION_COMMAND_RANDOM	EVENT_\
3	3			5	11-15	40	90	EXPERT	SIMULATION_COMMAND_RANDOM	EVENT_\
4	1	chainmodel 1	normative behavio	5	1-5	40	40	BEGINNER	SIMULATION_COMMAND_RANDOM	EVENT_\
4	2			5	6-10	40	50	BEGINNER	SIMULATION_COMMAND_RANDOM	EVENT_\
4	3			5	11-15	40	60	BEGINNER	SIMULATION_COMMAND_RANDOM	EVENT_\
5	1	chainmodel 1	normative behavio	5	1-5	40	40	ADVANCED	SIMULATION_COMMAND_RANDOM	EVENT_\
5	2			5	6-10	40	50	ADVANCED	SIMULATION_COMMAND_RANDOM	EVENT_\
5	3			5	11-15	40	60	ADVANCED	SIMULATION_COMMAND_RANDOM	EVENT_\
6	1	chainmodel 1	normative behavio	5	1-5	40	40	EXPERT	SIMULATION_COMMAND_RANDOM	EVENT_\
6	2			5	6-10	40	50	EXPERT	SIMULATION_COMMAND_RANDOM	EVENT_\
6	3			5	11-15	40	60	EXPERT	SIMULATION_COMMAND_RANDOM	EVENT_\
7	1	chainmodel 1	selection mode	19	1-19	40	70	BEGINNER	SIMULATION_COMMAND_RANDOM	EVENT_\
8	2			19	20-38	40	70	BEGINNER	SIMULATION_COMMAND_SIMPLE_CONSTRAINT	EVENT_\
9	1	chainmodel 1	selection mode	10	1-10	40	60	ADVANCED	SIMULATION_COMMAND_RANDOM	EVENT_\

Figure 6.1: Documentation of all simulation runs

The simulations are primarily used to evaluate the configurability concept and to reduce the number of possible final candidates (configurations) for the APOSDLE system.

The first sub chapter 6.1 describes the steps needed to be conducted for the simulation. The following sub chapter 6.2 reveals the results and findings of the conducted simulations.

6.1 Simulation steps

To conduct a simulation we need to clearly define all steps in the process of simulation. These steps have been made based on the general description of the simulation process (see sub chapter 4.1) and the system restriction made by APOSDLE system. Conjunction of these two types of requirements resulted in following simulation steps:

1. Formulating a problem space we want to research
2. Drawing the simulation design
3. Creating a trial description (see Figure 6.1): this is a document which contains overview over all simulation runs. It should include parameters relevant for a simulation run:
    - no (sim run): Number of simulation runs
    - no (sim unit): Simulation unit is an one set of simulation parameters executed by single user
    - Chan model: Describes the used Chain model (chain model 1 is a 14 Event chain model and chain model 2 is a 9 Event chain model)

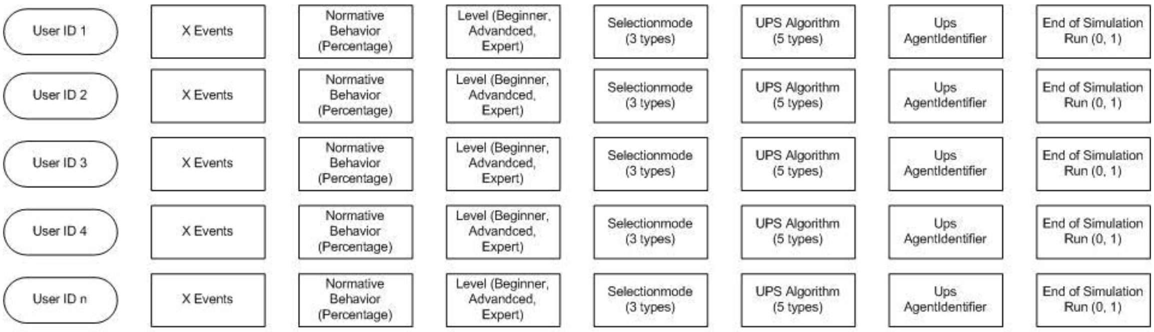
- Problem topic (research question)
- n (personas): Number of persons within this simulation unit
- User-ID
- Number of events
- Percentage of normative behavior: Describe the amount of normative (=expected, right) behavior
- Input experience level: Input level of a person (BEGINNER, ADVANCED, EXPERT)
- Selection mode event: The mode of simulation constraints in generating events for the simulation (SIMULATION_COMMAND_RANDOM, SIMULATION_COMMAND_SIM and SIMULATION_COMMAND_COMPLEX_CONSTRAINT)
- UPS Algorithm and Ups_Agentidentifier where the Ups_Agentidentifier is the name of the configuration of algorithm and chain model:
  - FREQUENCY -> FrequencyModel
  - FREQUENCY_RECENT_NUMBER_OF_EXECUTIONS ->FrequencyRecentNumExecu
  - WEIGHTING -> WeightingModel
  - WEIGHTING__RECENT_NUMBER_OF_EXECUTIONS -> WeightingRecentNumExecutionModel
  - EVENT_WEGHTING -> EventWeightingModel
  - EVENT_WEGHTING__RECENT_NUMBER_OF_EXECUTIONS -> EventWeightingRecentNumExecutionModel
- simulationTerminator: This is a flag which indicates the end of simulation run
- name (file): Defines the simulation file name

4. Creating a CSV formatted input file (see Figure 6.2)

5. Take notice of simulation constraints:

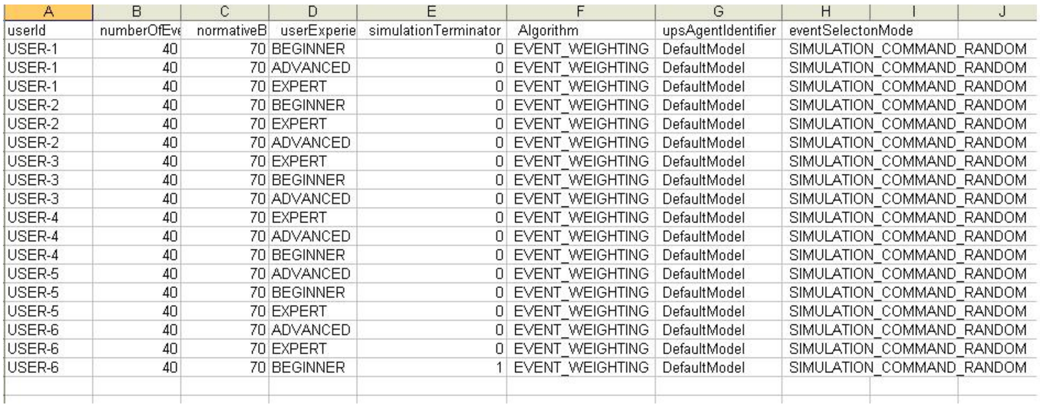
   - Don't simulation more than 19 users per simulation run (technical constraint)
   - When simulate a different behavior within a person, write separate lines for each behavior type (see Figure 6.3)

6. Starting the simulation run:



User ID 1	X Events	Normative Behavior (Percentage)	Level (Beginner, Advandced, Expert)	Selectionmode (3 types)	UPS Algorithm (5 types)	Ups AgentIdentifier	End of Simulation Run (0, 1)
User ID 2	X Events	Normative Behavior (Percentage)	Level (Beginner, Advandced, Expert)	Selectionmode (3 types)	UPS Algorithm (5 types)	Ups AgentIdentifier	End of Simulation Run (0, 1)
User ID 3	X Events	Normative Behavior (Percentage)	Level (Beginner, Advandced, Expert)	Selectionmode (3 types)	UPS Algorithm (5 types)	Ups AgentIdentifier	End of Simulation Run (0, 1)
User ID 4	X Events	Normative Behavior (Percentage)	Level (Beginner, Advandced, Expert)	Selectionmode (3 types)	UPS Algorithm (5 types)	Ups AgentIdentifier	End of Simulation Run (0, 1)
User ID n	X Events	Normative Behavior (Percentage)	Level (Beginner, Advandced, Expert)	Selectionmode (3 types)	UPS Algorithm (5 types)	Ups AgentIdentifier	End of Simulation Run (0, 1)

Figure 6.2: Input file as CSV



A	B	C	D	E	F	G	H	I	J
userId	numberOfEve	normativeB	userExperie	simulationTerminator	Algorithm	upsAgentIdentifier	eventSelectonMode		
USER-1	40	70	BEGINNER	0	EVENT_WEIGHTING	DefaultModel	SIMULATION_COMMAND_RANDOM		
USER-1	40	70	ADVANCED	0	EVENT_WEIGHTING	DefaultModel	SIMULATION_COMMAND_RANDOM		
USER-1	40	70	EXPERT	0	EVENT_WEIGHTING	DefaultModel	SIMULATION_COMMAND_RANDOM		
USER-2	40	70	BEGINNER	0	EVENT_WEIGHTING	DefaultModel	SIMULATION_COMMAND_RANDOM		
USER-2	40	70	EXPERT	0	EVENT_WEIGHTING	DefaultModel	SIMULATION_COMMAND_RANDOM		
USER-2	40	70	ADVANCED	0	EVENT_WEIGHTING	DefaultModel	SIMULATION_COMMAND_RANDOM		
USER-3	40	70	EXPERT	0	EVENT_WEIGHTING	DefaultModel	SIMULATION_COMMAND_RANDOM		
USER-3	40	70	BEGINNER	0	EVENT_WEIGHTING	DefaultModel	SIMULATION_COMMAND_RANDOM		
USER-3	40	70	ADVANCED	0	EVENT_WEIGHTING	DefaultModel	SIMULATION_COMMAND_RANDOM		
USER-4	40	70	EXPERT	0	EVENT_WEIGHTING	DefaultModel	SIMULATION_COMMAND_RANDOM		
USER-4	40	70	ADVANCED	0	EVENT_WEIGHTING	DefaultModel	SIMULATION_COMMAND_RANDOM		
USER-4	40	70	BEGINNER	0	EVENT_WEIGHTING	DefaultModel	SIMULATION_COMMAND_RANDOM		
USER-5	40	70	ADVANCED	0	EVENT_WEIGHTING	DefaultModel	SIMULATION_COMMAND_RANDOM		
USER-5	40	70	BEGINNER	0	EVENT_WEIGHTING	DefaultModel	SIMULATION_COMMAND_RANDOM		
USER-5	40	70	EXPERT	0	EVENT_WEIGHTING	DefaultModel	SIMULATION_COMMAND_RANDOM		
USER-6	40	70	ADVANCED	0	EVENT_WEIGHTING	DefaultModel	SIMULATION_COMMAND_RANDOM		
USER-6	40	70	EXPERT	0	EVENT_WEIGHTING	DefaultModel	SIMULATION_COMMAND_RANDOM		
USER-6	40	70	BEGINNER	1	EVENT_WEIGHTING	DefaultModel	SIMULATION_COMMAND_RANDOM		

Figure 6.3: Behavior modification input file as CSV

- Use the template for creating a input file (input parameters draft)
- Save the file in consideration of the following points:
  - number of personas: "*nxx*"
  - number of events:"*eventxx*"
  - normative behavior:"*nb70_80_90*"
  - expertise level:"*beg_adv_exp*"
  - selection mode:"*sm1_2_3*"
  - algorithm:"*alg1_2_3_4_5*"

Example: n10_event40_nb70_80_beg_sm1_2_alg1

This means 10 persons, each with 40 events, normative behavior 79, 80 and 90%, Beginner level, Selection mode 1 and 2, and using algorithm 1-FREQUENCY)

- Start the simulation framework (application) and select the input file and the folder where the generated output file will be saved

7. Import the CSV-formatted output file into SPSS[1]

After following these steps the result (output) should be in form of SPSS graphics. This makes it easier to visually compare different configuration. The following sub chapter describes some of the simulation conducted which had most impact on the decision making process of which configuration to use in APOSDLE.

6.2 Simulations with APOSDLE P3 User Profile Service

This sub chapter describes the most relevant parts of the simulation process. The intention of this thesis is not to present all simulation results but the ones which had most impact on the decision, which configuration to use in the current APOSDLE system.

First step in simulation process was to identify the problem space and determine the research questions:

- Are there differences in detecting the knowledge level (BEGINNER, ADVANCED or EXPERT[2]) between the percentages of *normative behaviors*[3]?

[1] SPSS http://www.SPSS.com/

[2] These are the levels which indicates the knowledge of the user in the system based on his/hers behavior in the system. For example the BEGINNER persons are often requesting assistance from the system or other knowledgeable persons in system. Such knowledgeable person would be EXPERT in this case.

[3] Normative behavior means the extent to which simulated user behavior corresponds to the simulated

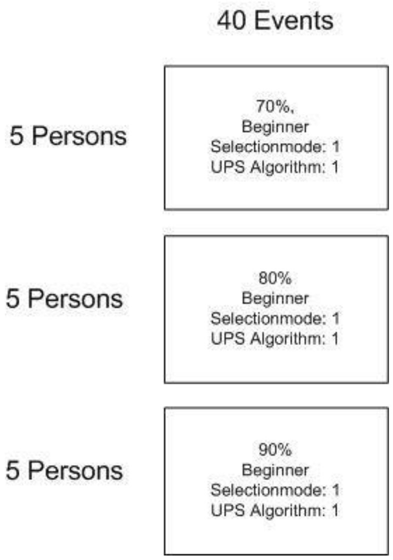


Figure 6.4: Different amounts of Beginner normative behavior

- How many simulation steps (events) do algorithms need to identify the correct (simulated) level?
- How long does it take for different configurations (algorithm plus chain model) to detect modifications in the user level?
- Is there a difference in the correctness of inferences when using different chain models (with the same algorithm) in the inferring process?
- Which algorithm shows the best results for the given chain model? In other words, are there differences between different algorithms when using the same chain model?

6.2.1 Differences in detecting the knowledge level between the percentage of "normative behaviors"

After conducting a series of simulations with different knowledge level types (see Beginner example Figure 6.4) it was obvious that for the correct inferring, system needs at least 60 % of normative behavior regardless to chain model and algorithm (see Figure 6.5). This means, if 60% of a user's activities are related to one of the three levels, APOSDLE will be able to detect that level. This result was expected and confirmation that our inferring algorithms are doing their job correctly.

The results for the other two cases (Advanced and Expert) were very similar.

knowledge level

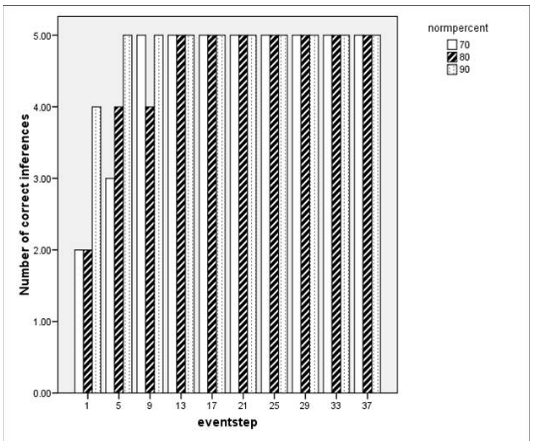


Figure 6.5: Result of simulation for the different amount of Beginner normative behavior

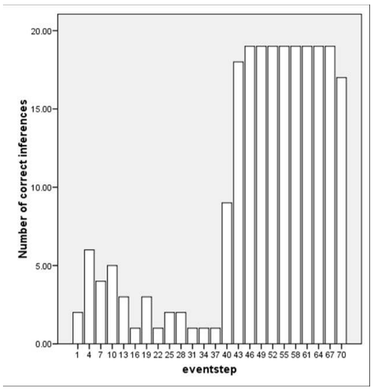


Figure 6.6: Level detection for the EventWeightingAlgorithm

6.2.2 How many steps do algorithms need to converge to the right level

Simulations showed that, given 60% of normative behavior, on average, the algorithms need 40-50 events to make a correct inference. The answer to the question of how many steps do algorithms need to converge to the right level provides us with the answer to how many recent executions do we need to have for the RECENT_NUMBER_OF_EXECUTIONS type algorithms (see sub chapter 5.4): Because on average, the algorithms need 40-50 events to make a correct inference (see Figure 6.6), also the number of events that should be the basis for the inference should be limited to the past 40-50 events.

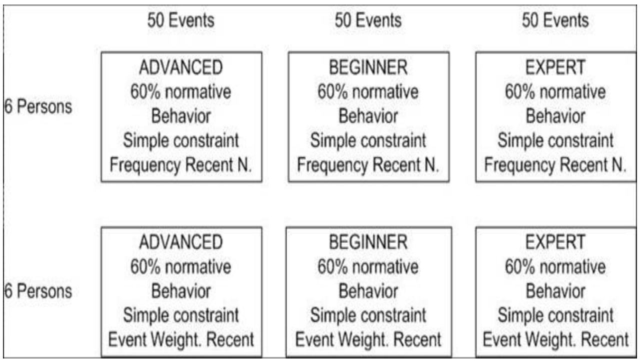


Figure 6.7: Modification of user behavior

6.2.3 Are there differences in detecting the right level within a sequence by changing the level and how long does it take for different configurations (algorithm plus chain model) to detect modifications in the user level

Here we tried to simulate the learning process of a user, meaning that the user actions are changing over time as a consequence of learning (Beginner->Advance->Expert) or knowledge aging (Expert->Advanced->Beginner) process (see Figure 6.7). The goal was to compare the two types of the inferring algorithms (DEFAULT vs. RECENT_NUMBER_OF_EXECUTION

It was obvious that the DEFAULT type algorithms are having trouble inferring correctly when the behavior of the user is changing over time (see Figure 6.8). The reason for that was that the records about the previous behavior had too much influence on the inferring algorithms. As the figure 6.8 shows the EventWeighting Algorithm was not able to recognize the ADVANCED level user behavior at all (the gap between 50 and 140 events) and had problems detecting the EXPERT level as well.

After applying the EVENT_WEIGHTING_RECENT_NUMBER_OF_EXECUTIONS algorithm the problem was salved. This algorithm used last 40 events recorded for the inferring purposes and provided very good result (see Figure 6.9). He needed about 35-40 events to infer correct knowledge level of the user.

6.2.4 Difference between chain models in respect to inferring results

Two different chain models were used in simulations. The obvious question was which one of those two is better suitable for the APOSDLE (see sub chapter 5.3). The first model included 14 KIE whereas the second one comprised only 9. They were differing not only in number of KIE but also in terms of their mapping to levels. The difference

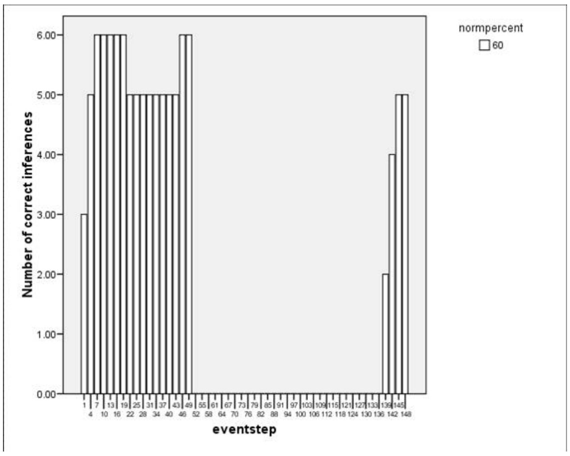


Figure 6.8: EventWeighting algorithm with user behavior changing

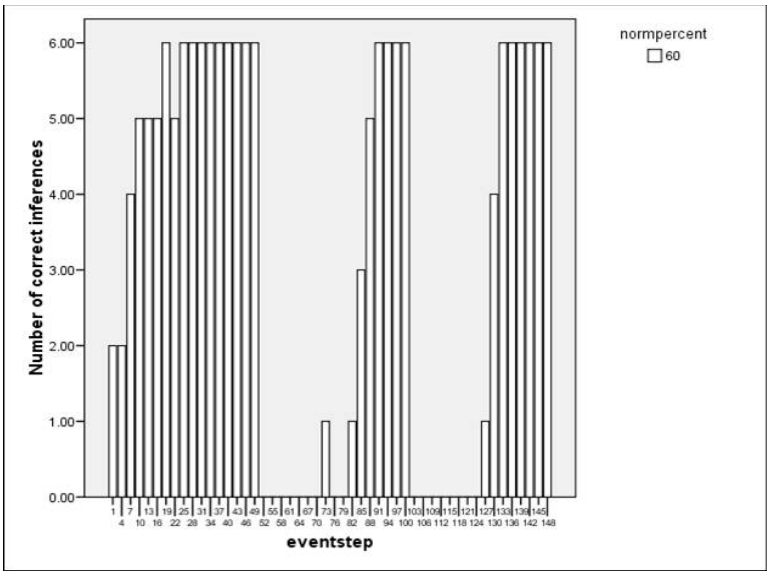


Figure 6.9: EventWeightingRecnetNumberOfExecutions algorithm with user behavior changing

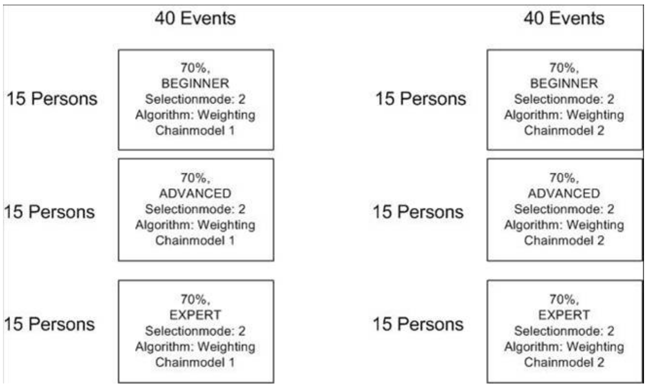

40 Events
40 Events
15 Persons
70%, BEGINNER Selectionmode: 2 Algorithm: Weighting Chainmodel 1
15 Persons
70%, BEGINNER Selectionmode: 2 Algorithm: Weighting Chainmodel 2
15 Persons
70%, ADVANCED Selectionmode: 2 Algorithm: Weighting Chainmodel 1
15 Persons
70%, ADVANCED Selectionmode: 2 Algorithm: Weighting Chainmodel 2
15 Persons
70%, EXPERT Selectionmode: 2 Algorithm: Weighting Chainmodel 1
15 Persons
70%, EXPERT Selectionmode: 2 Algorithm: Weighting Chainmodel 2

Figure 6.10: Chain model comparison

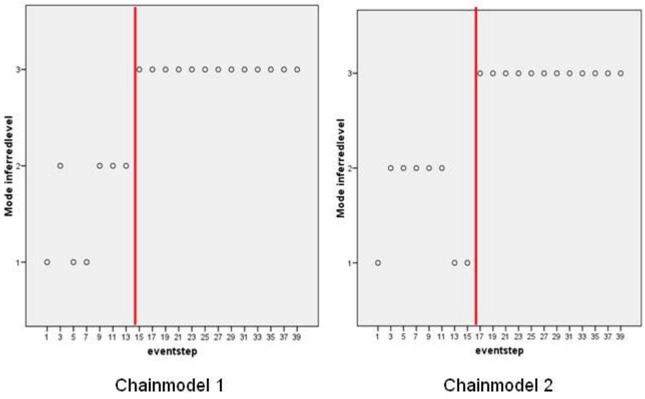


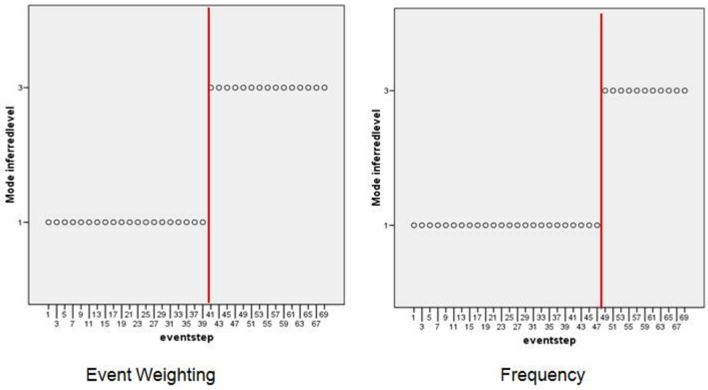
Figure 6.11: Chain model comparison with Weighting algorithm

in mapping was relatively small and the main question was if the number of KIE could be reduced so that the inferring process is simplified, without losing the preciseness of inferring (see Figure 6.10).

After conducting a simulation with the Weighting algorithm it was obvious that although the reduced Chain model needed few extra steps to make correct inferring, it was precise enough (see Figure6.11).

6.2.5 Which algorithm shows the best results for the given chain model

At the end there was only one question remaining. Which algorithms are best suitable for the APOSDLE domains? After conducting a numerous simulations with different configurations we managed to narrow down the number of candidate configuration (algorithm plus chain model) for the UPS inference process. After seeing that DEFAULT type



Event Weighting

Frequency

Figure 6.12: Comparison between EventWeightingRecentNumberOfExecutions and FrequencyRecentNumberOfExecutions

inferring algorithms are not good for inferring a user knowledge level in cases when user changes his normative behavior, it was obvious that the final solution would be one of the RECENT_NUMBER_OF_EXECUTIONS type algorithms. And indeed, we managed to distinguish two of them. These are: FREQUENCY_RECENT_NUMBER_OF_EXECUTIONS and EVENT _WEGHTING_RECENT_NUMBER_OF_EXECUTIONS algorithm (see Figure 6.12).

With very small differences, both algorithms showed good results. Therefore these two algorithms were seen as top candidates for the final solution.

6.3 Conclusion

The whole configurability concept was always threatening to explode in complexity. Therefore one of the first principles in decision making process was to try to reduce the complexity as much as possible.

The clear results of the simulations were four configurations. Both chain models showed good results and the best algorithms are FREQUENCY and EVENT_WEIGHTING algorithm from RECENT_NUMBER_OF_EXECUTIONS type. Led by our principle of choosing the simplest solution, the 9-Event Chain model and the simplest algorithm, namely FREQUENCY_RECENT_NUMBER_OF_EXECUTIONS was given the preference.

Even though the generalizability of the outcomes is limited by the fact that a simulation can never replace a real user study, we still believe that the simulation provides us with answers to the very basic issues. It is possible when the real usage data comes, that some other (more complex) configurations may lead to better inferences.

This also should be seen as the first step in component evaluation of the UPS component. The evaluation will be done in three phases: phase one simulation, phase two labor experiment, and phase three the field experiment.

7 APOSDLE technical view

This chapter provides the general overview over the most important components of the APOSDLE system. APOSDLE has a typical client-server architecture realized in java technologies. The number of different frameworks such as Spring[1] , Hibernate[2] , and Axis2[3] were used to develop the software. The client communicates with different services provided by the APOSDLE Platform (see Figure 7.1). Services are comprised of conceptually similar functions and are defined in WSDL format. The components on the client side communicate with the APOSDLE platform using clearly defined interface. APOSDLE is a highly complex system and the goal of this chapter is to explain the most relevant services on the platform side which are directly accessed by the client components. In the following, each of the APOSDLE Platform components and their relationship to APOSDLE client components will be described. But first a short overview over the most relevant client side components.

7.1 Client Side Components

There are different types of Knowledge Resources accessible to the APOSDLE user: Tasks, Topics, Learning Paths, Documents, Snippets, Cooperation Transcripts and Persons. All these resources can be organized into so called Collections and shared throughout the APOSDLE and thus may be considered as Knowledge Resources themselves. The role of the client side components is to provide the access to the user to all these knowledge resources. To do so user can use different client components:

- *APOSDLE Suggest and Search* are basically the main entrance points where knowledge resources can be accessed. Search uses the text based search capabilities for retrieving the knowledge resources, while APOSDLE Suggest has more intelligent approach, namely using the intelligent algorithms for retrieving the knowledge resources relevant to the user's current context of work.

[1] Spring http://www.springsource.org/

[2] Hibernate https://www.hibernate.org/

[3] Axis2 http://ws.apache.org/axis2/

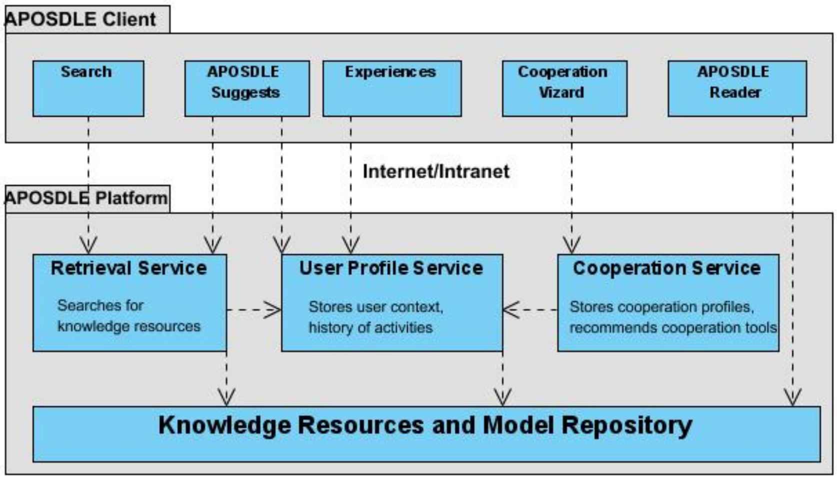

APOSDLE Client
Search
APOSDLE Suggests
Experiences
Cooperation Vizard
APOSDLE Reader
Internet/Intranet
APOSDLE Platform
Retrieval Service
Searches for knowledge resources
User Profile Service
Stores user context, history of activities
Cooperation Service
Stores cooperation profiles, recommends cooperation tools
Knowledge Resources and Model Repository

Figure 7.1: APOSDLE Client-Server Architecture

- Experiences is a direct access to the user related data. This is the visual representation in a treemap[4] visualization method of the user profile in the system. The user can see all topics and their mappings to the knowledge levels as well as the action which led to this decision.

- Cooperation Wizard is a component which helps the users during their cooperation processes. It makes the cooperation structure visible and provides the transcripts of the cooperation.

- APOSDLE Reader is a central access to Snippets and Documents. It is based on the ADOBE PDF viewer and a video player for all standard audiovisual formats. In order to see the documents, APOSDLE converts them into .pdf files. These documents cannot be modified. On the other hand the APOSDLE Reader offers an innovative approach in annotating these documents and thus creating so called snippets. The user can select the part of the document and relate him to a particular topic. This snippet is later recognized as a resource in the APOSDLE system.

These are not all client components but the ones with the most impact in the system. The next sub chapters are describing the components on the platform side, which are supporting these client components and providing them the intelligence in the process of retrieving the knowledge resources.

[4] treemap: http://www.cs.umd.edu/hcil/treemap/

7.2 User Profile Service (UPS)

The User Profile Service is responsible for the adaptivity of the system and is therefore a central component in the system. Almost each component on the client side depends directly or indirectly from this service. APOSDLE Suggest depends almost entirely on UPS. The Task, Topic and Learning Paths are retrieved by the UPS. The UPS also ranks the Learning Goals based on the learning gap of the user. Retrieval of the knowledge content such as Snippets, Documents etc. is combined effort of UPS and RS. Experiences is a visual representation of the User Profile and therefore directly dependent on UPS. UPS provides the information not only about the current level of knowledge a user has acquired for a particular Topic, but also the list of user activities used to infer this level. Other services can make use of the data stored in the UPS and query this data for their purposes.

7.3 Retrieval Service (RS)

The Retrieval Service is a general service responsible for retrieval of Knowledge Resources. It is used by Search and APOSDLE Suggests component on the client side. APOSDLE Search uses RS for full text search in a Lucene -based search engine. APOSDLE Suggest on the other hand uses RS together with UPS and an associative network to find Knowledge Resources similar to the current user's context.

7.4 Cooperation Service (CS)

The Cooperation Service is responsible for the cooperation activities within the APOSDLE system. This service also maintains the reflections made by users about their cooperation. CS uses UPS to provide the context for the cooperation and in return CS provides the UPS with valuable data relevant for maintaining user profiles. The Cooperation Wizard client component uses the CS to initiate cooperation, share context information, and create notes during a cooperation.

7.5 Knowledge Resources and Model Repository (KRMR)

The Knowledge Resources and Model Repository is a repository where different data models are maintained. It is a common interface, an abstraction layer to distributed underlying (data) models. These models could be representation of working processes in a company abstracted as ontology model, or access to different resources such as snippets, documents, and multimedia stored in different data repositories. This service is used by all other platform services and therefore indirectly by all client components. APOSDLE Reader retrieves all Knowledge Resources directly from the KRMR to visualize them.

The following chapter describes the UPS components from configurability point of view in more detail.

8 Architecture and Components of UPS from Configurability perspective

This chapter describes the components as well as the architecture of the UPS P3 (User Profiling Service Prototype 3) from configurability point of view. The User Profile Service (UPS) is the software component, which maintains the user profiles of APOSDLE users and provides different user related services (e.g., document recommender service, people recommender service) to the other parts of APOSDLE (client and server). In a nutshell, the UPS has two goals: First, the diagnosis of a user's knowledge level (beginner, ADVANCED, expert) in each topic in the learning domain, and second, the diagnosis of changes in the knowledge level (maintenance).

First sub chapter 8.1 describes the UPS services in general. The intention is to get a picture about the UPS as a bundle of services used by the rest of the system. The chapter about APOSDLE (see chapter 7) shows that the UPS is a central and one of the most important components of the APOSDLE system. UPS is also one of the biggest and richest (in respect to functionalities) service throughout the whole system (not just on the platform side). It is also the best documented and tested service in system. Due the many dependencies to this service it was necessary to pay more attention on the quality of the code during the design and developing phase. If the UPS does not work properly the whole platform is not functional and therefore the APOSDLE system in general. Configurability as underlying concept of the UPS is present in almost each of the components. The following sub chapters are describing these components and their relationship to the configurability approach. Due the importance and the central role of UPS in the APOSDLE system this concept has a great impact on the whole system as well.

The most important parts of the UPS are (see Figure 8.1):

- UPS Services (see sub chapter 8.1)
- Data Model (see sub chapter 8.2)
- User Manager (see sub chapter 8.3)

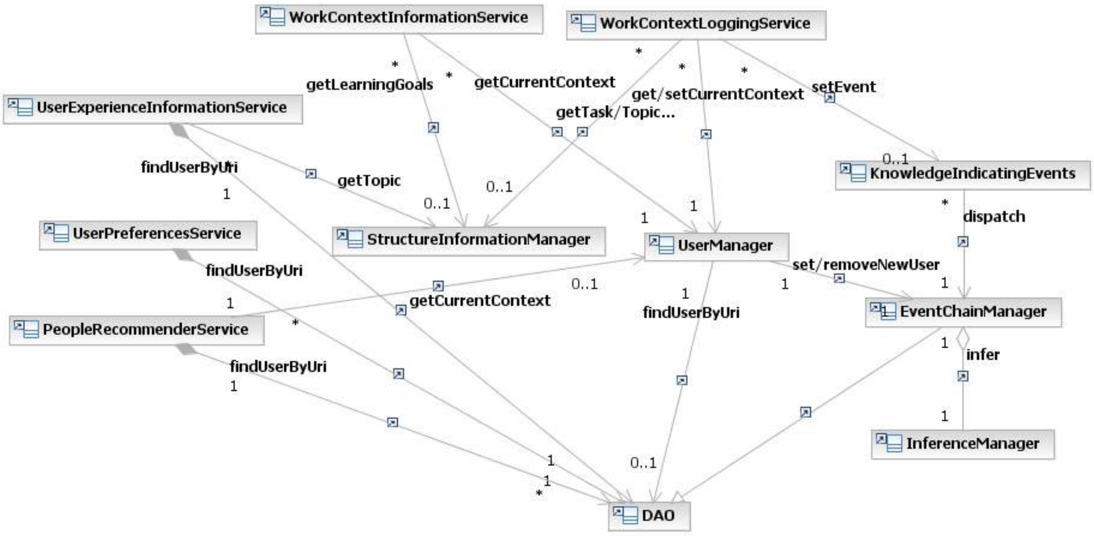


Figure 8.1: UPS Architecture simple view

- Event Chain Manager (see sub chapter 8.4)
- Inference Manger (see sub chapter 8.5)

All these parts are described in following sub chapters.

8.1 UPS Services

As shown in figure 8.1 the UPS consist out of many services. They are providing the rest of the system with personalized data prepared by the UPS. The configurability approach is not present in these services explicitly but implicitly trough other components. These components are providing the underlying infrastructure for the services to work upon them. The services present in UPS are:

- WorkContextInformationService: provides the direct link to underlying ontology models and returns the requested Task/Topics/Learning Paths/Learning Goals. These resources are personalized if so needed. For example the returned list of the Learning Goals is sorted/ranked based on the user profile.
- WorkContextLoggingService: is used to set the current context of user for the further purposes. This context is used by various services like cooperation, WorkContextInformationService, UserPreferenceService etc...

- UserExperienceInformationService: is predominantly used by the Experiences component on the client side. But also APOSDLE reader and Privacy Service are using the information provided by this service. Experiences holds the entire user profile for the visualization purposes. APOSDLE Reader and Privacy Service are interested in the last ten topics a particular user worked in.

- UserPreferenceService: as the name of the service says this service is responsible for saving the users preferences. It also used by other services to keep track of the documents uploaded by a user.

- PeopleRecommenderService: is a service called by the RetrievalService for recommending users relevant for a particular topic. The relevant users are recommended based on their user profiles.

All above stated services are not just using the information prepared by UPS, but also providing the valuable data for the personalization purposes. The UPS collects these data and stores them in a very flexible underlying data model. The following sub chapter describes this data model.

8.2 Data Model

Data model is very important for the whole configurability concept. Only if the data model is flexible enough we can pursuit the flexible approach the configurability is offering. In order to provide such data model I used hibernate top-down approach. The newest trends and technologies in so called ORM approach are used to create the data model represented as an object abstraction layer, and to create the data base schema to support and persists the information contained in those objects. The advantage is that the whole work in creating the data base schema is left for the Hibernate to deal with. Therefore the focus could be shifted to some more important things such creating a flexible, scalable and extendable data model. The objects and thus the data model created is showed in figure 8.2.

The lowest construct is the Event. The Event can have different types. They are all hierarchically ordered where the children are inheriting the parent properties. There are numerous information saved in Event objects:

- The time stamp

- Topic (context) under this event is created

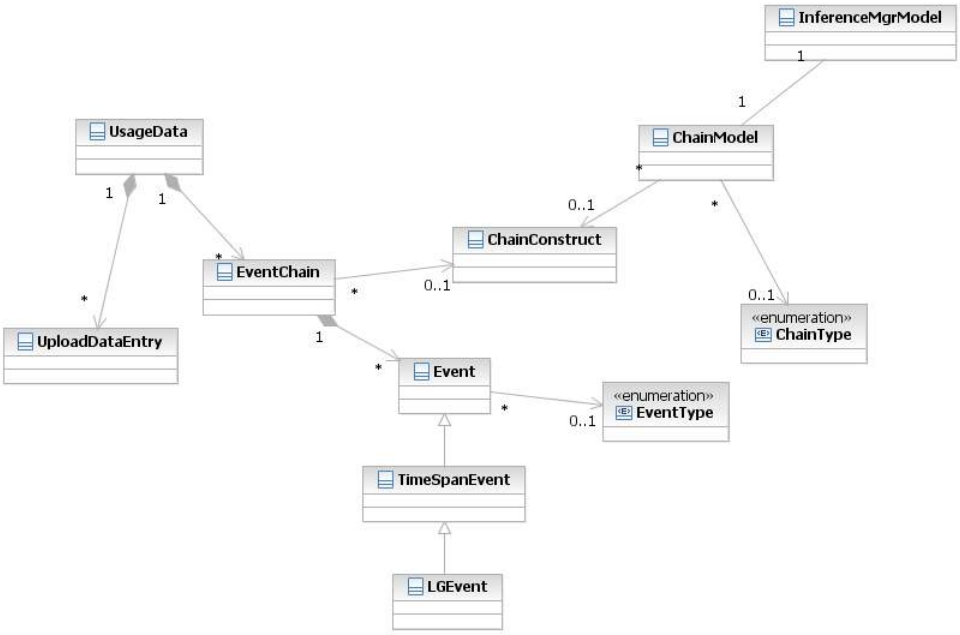


Figure 8.2: Data Model of the UPS

- The Id of the user which created this event
- And a lot of different information relevant to the event type

The Event is used as a part of the Event Chains, or other formulated the Event Chains are collection of Events.

Event Chains are used as a part of Usage Data and the Chain Constructs. Usage Data are using the Event Chains for the fast access to user activities in system. The Usage Data holds all data relevant to a user in UPS.

Chain Constructs are something like a constructs where the Event Chains are their instances. Here is the structure of the Chain Models (see sub chapter 5.3) preserved.

The Chain Model object holds references to a different Chain Constructs and maintains the chain models present in the system.

Inference Manager Model holds the configurations present in the system. This object has a reference to the Chain Model and the information about which Algorithms are present in the system.

8.3 User Manager

User Manager has a duty to maintain the logged users in a system. If the user is not logged-in this component restricts the access to UPS data. Just reduced set of user data are allowed. Although seems simple, the activities of this component are very complex. This component manages and synchronizes the multi threading in UPS and thus preventing the common errors caused by multi threading. At moment the configurability approach is not present in this component. But in time and further development it is possible that this component has a control of which configuration to switch based on the user type. This fine granular usage of the configurability is not yet present in the system.

8.4 Event Chain Manager

Event Chain Manager is a first intelligent component in the system. This component deals with the creation of the events based on the policies defined for this purposes. These policies are pre-defined by the Chain Models structure present in configuration. This component looks into data base for the right Chain Model structure and based on that creates the policies used for the filtering of the incoming events. The maintaining the chain models, is also done by this component. It tracks the user activities and tries to filter the dirty data out of the system. The dirty data are all those user actions performed without context and purpose (meaning just clicking around). After an Event had passed all checks made by this component, he could be handed over to inferring mechanism and persisted into data base. This component depends entirely on the preselected configuration. If the configuration for any reason is not pre selected, Event Chain Manager cannot operate.

8.5 Inference Manager

Inference Manager is a second intelligent component in the system. After one Event had passed all checks made by Event Chain Manager, the Inference Manager is sure that the Events he gets are correct and ready for inferencing. Inference Manager holds the right configuration from Data Model and based upon this configuration sets the right inferring algorithm to be used for inferring purposes. After this process is finished the newly acquired information is persisted into usage data. Hence the user profile is updated and ready for the exploitation. Again without information from configuration, Inference Manager cannot be initialized with the correct algorithm and therefore cannot proceed

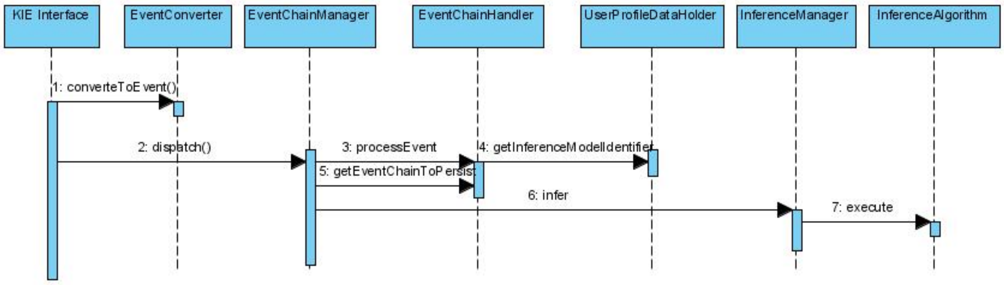


Figure 8.3: Data flow inferring process

with inferring activities.

8.6 Data Flow

As described in previous sub chapter the user profile is updated each time a user fires an event in system. This way we can be sure that the user profile is up to date. What exactly happens when one event comes in UPS is presented in figure 8.3.

As figure 8.3 shows the Event comes into UPS over KIE interface. This is the entry gate for all incoming events. First the events need to be properly structured for the UPS purposes. This is done by EventConverter. After Event object is made, he will be turned over to the EventChainManager for the further processing. So far Event object is nothing more but the hollow shell ready to be filled in and having only information about the resource from which this event was fired. EventChainManager sets the correct chain model structure from configuration and gives the Event to the EventChainHandler which checks if this Event satisfies all pre conditions. If this is the case this event is injected into EventChain. After EventChainHandlers informs the EventChainManger that this Event had passed all checks he provides the Event with the context elements. These are the Topic information which the underlying resource in event has relation to. After this checks and enrichment is finished the Event is given to the InferenceManager for inferring. InferenceManager selects the appropriate Inferencing Algorithm out of the configuration and executes him over the recorded events. The result of this inferring process will be persisted and user profile will be updated.

8.7 Conclusion

The UPS is one of the most complex and important components throughout the entire APOSDLE architecture. The UPS itself is constructed in a very modular and maintainable way. Each of the components implemented and integrated into UPS is made with respect to the most modern and important software engineering developing paradigm. The configurability approach is present in almost each component of the UPS. Especially the components responsible for the intelligence such as Event Chain Manager or Inference Manager are entirely dependent of this approach. These components are using the information from configuration to choose the right chain models structures and algorithms, which are later used for the filtering and inferring purposes. Extensive testing and documentation made this service suitable for the further development and maintenance purposes.

9 Simulation framework

One way to prove if the user profile service is correct is to perform simulations on UPS. The goal is to try to find out the best combination or combinations of chain models and algorithms, depending on domains and person behavior types(see chapter 6).

The first challenge was to extract the UPS out of APOSDLE system in order to conduct simulations with this service. Although UPS stands for the User Profiling Service it is not a service in a technical sense. This service is rather a component or a module on the APOSDLE platform side than a real web service. Therefore it was necessary to adapt this component in order to become a real autonomous service. After being extracted, the UPS was wrapped and prepared to be used by external simulation framework developed for the simulation purposes. The design and the components of this framework are described in the sub chapter 9.1. The data flow is described in the sub chapter 9.2. Sub chapter 9.3 describes the component implemented for the evaluation purposes when the real usage data comes.

9.1 Simulation framework design

The architecture of the simulation framework is shown in (see Figure 9.1). The most distinguished parts are:

- Input Handling
- Event (Pool) Creator
- Simulation Commands
- Writer

These parts are explained in detail in following sub chapters.

9.1.1 Input Handling

Input Handling part deals with the different input types. The intention was to have different input media possibilities such as the console, graphical user interface or the

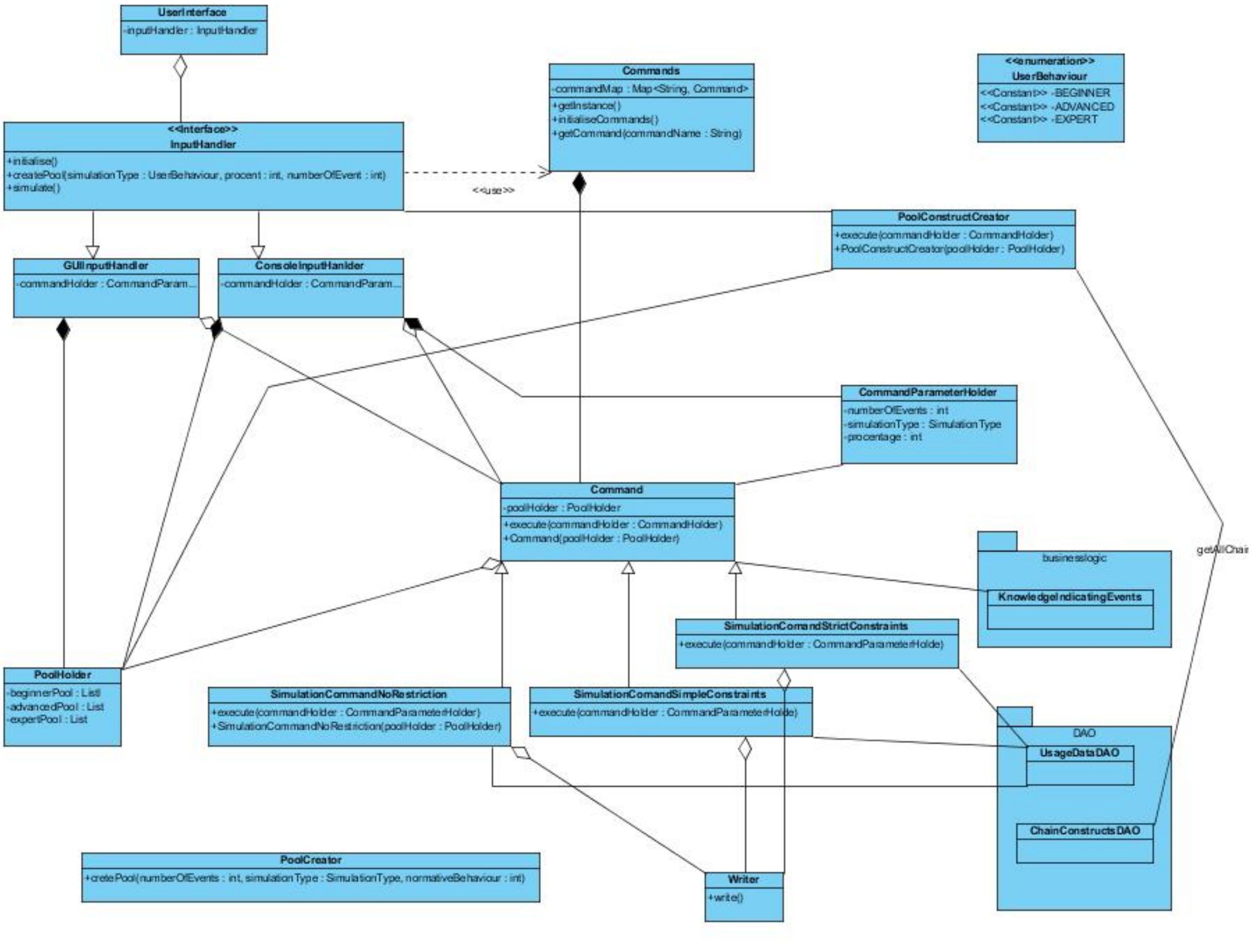

UserInterface
-inputHandler : InputHandler
<<Interface>>
InputHandler
+initialise()
+createPool(simulationType : UserBehaviour, procent : int, numberOfEvent : int)
+simulate()
<<use>>
Commands
-commandMap : Map<String, Command>
+getInstance()
+initialiseCommands()
+getCommand(commandName : String)
<<enumeration>>
UserBehaviour
<<Constant>> -BEGINNER
<<Constant>> -ADVANCED
<<Constant>> -EXPERT
GUIInputHandler
-commandHolder : CommandParam...
ConsoleInputHanlder
-commandHolder : CommandParam...
PoolConstructCreator
+execute(commandHolder : CommandHolder)
+PoolConstructCreator(poolHolder : PoolHolder)
CommandParameterHolder
-numberOfEvents : int
-simulationType : SimulationType
-procentage : int
Command
-poolHolder : PoolHolder
+execute(commandHolder : CommandHolder)
+Command(poolHolder : PoolHolder)
businesslogic
KnowledgeIndicatingEvents
getAllChai
SimulationComandStrictConstraints
+execute(commandHolder : CommandParameterHolde)
PoolHolder
-beginnerPool : Listl
-advancedPool : List
-expertPool : List
SimulationCommandNoRestriction
+execute(commandHolder : CommandParameterHolder)
+SimulationCommandNoRestriction(poolHolder : PoolHolder)
SimulationComandSimpleConstraints
+execute(commandHolder : CommandParameterHolde)
DAO
UsageDataDAO
ChainConstructsDAO
PoolCreator
+cretePool(numberOfEvents : int, simulationType : SimulationType, normativeBehaviour : int)
Writer
+write()

Figure 9.1: Simulation framework architecture

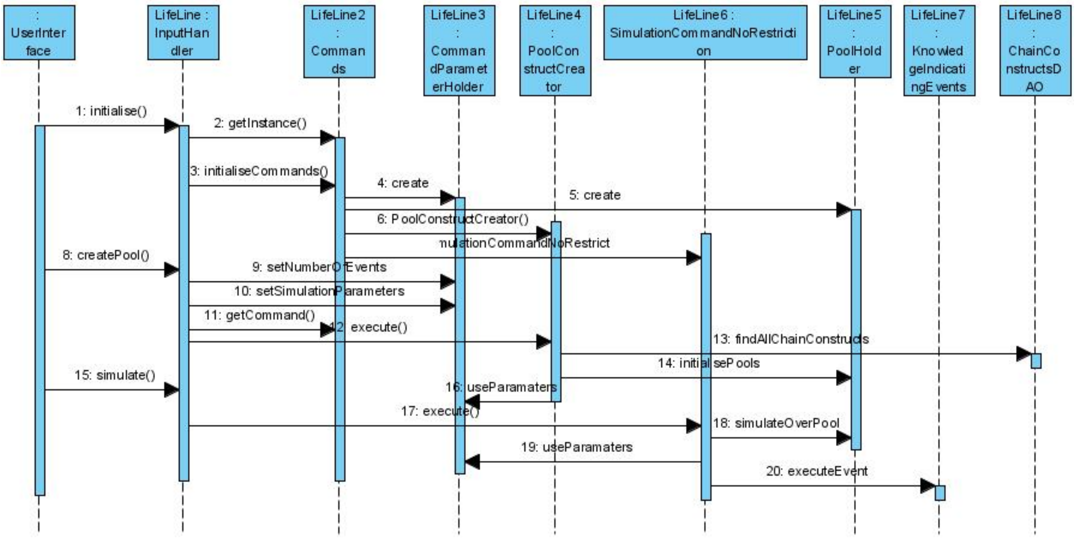


Figure 9.2: Sequence Diagram for simulations

input file (.csv file type). This should make this component more flexible and extendable for the different usage scenarios. The requirement was to have flexible input media and still not to invest to much resources in implementing this part. After analyzing the requirements and the needs of the simulation the final decision was to use the input file in .csv format as s input media for user interface. This made it possible to define and run several simulation runs at the same time without having to start each simulation run manually.

9.1.2 Event (Pool) Creator

Event (Pool) Creator is in charged for the creation of the predefined number of events which are to be used in simulation. It is important to differentiate between Event Constructs and Events (instances). The number of Event Construct is limited where the number of Events (instances) is unlimited. The process of creating the events for the simulation is divided into two parts.

First the Event Constructs are extracted from the Chain Model (defined in configuration). This means if the Chain Model is build from 14 Event Constructs this pool will hold these 14 Event Constructs. After extracting the Event Constructs into Event Construct Pool, the Pool Creator divide this pool into three smaller pools holding BEGINNER, ADVANCED and EXPERT Event Constructs respectfully.

Secondly when the simulation is started, two things have to be defined: the number of the events to be used in simulation, and the percent of the normative behavior. For example if the number of the Events is set to 100 and the normative behavior for BEGINNER is set to 70% then the 70 Events (instances) are created out of the BEGINNER Events Construct Pool and the rest of the (30) events from the other two pools. So created Events are than saved by the Simulation Command and used for the real simulation.

9.1.3 Simulation Commands

Simulation Commands are the implementation of the simulation runs based on the restriction level. There are random, simple constraint and complex constraint commands. These commands are operating over the event pool and deciding which events in which order should be taken into simulations. This ordering of the events depends from the restriction level. For the random type it is not relevant at all, meaning that the events are simply retrieved from the event pool and directly passed over to the UPS. In simple constraint command the ordering is very important. This means that certain events are having some precondition which have to be considered. For example if the user wants to edit a snippet he has to open his resource viewer first. The complex constraint command does the retrieval of the Events not programmatically but uses the list of manually compiled Events (for more details about the command restrictions see chapter 6).

Regardless of which way is used to retrieve Events, all these events are then passed to the UPS service where the Simulation Command supervise the execution and the outcome of the inferring process. The Simulation Commands are also in charge of creating the output files trough Writer.

9.1.4 Writer

The *Writer* creates the output files in a predefined form. So generated files are then used by the SPSS application for the graphical representation of the simulation results. Therefore the format of this file very important (for more details about the format see chapter 6). For the creation of the .csv files I used the special library, namely supercsv[1]. This library makes it possible to create complex and standardized .csv files.

Writer optimizes the writing effort by trying to reduce the number of actual writings. To do so he has to know when the simulation runs are finished and then try to write them at once in the .csv file. If there are too many data to hold in memory he writes the data in chunks and so optimize the writing actions.

[1] supercsv: http://supercsv.sourceforge.net/

9.2 Data Flow

The sequence diagram (see Figure 9.2) shows the sequence of the calls made during a simulation run. There are basically three main steps in each simulation:

- Initialization
- Event Pool creation
- Simulation

9.2.1 Initialization

Initialization does the initialization of the framework, meaning that all relevant data from the input file are read and imported into framework for the further purposes. Such data are: Number of Events (steps), User Id, Configuration to use, Algorithm, Simulation Command Type etc... The initialization process also extracts the events from the chain models and creates the pool of the event constructs which are later used for the creation of the events used in simulation. These are the steps from 1 to 7 in the sequence diagram (see Figure 9.2).

9.2.2 Event Pool Creation

Event Pool creation in respect to the selected simulation command (restriction mode) and the number of the events which should be used in simulation (defined in input file), creates the pool of the event objects (fully initialized). Steps from 8 to 14 (see Figure 9.2).

9.2.3 Simulation

Simulation is an actual simulation run. This is conducted by simulation command. It uses the created pools of the events and start sending the events into UPS and observing the inferring process. The output results are saved into output file for further processing by SPSS application.

9.3 Evaluation Component

The purpose of the simulation framework is to simulate the real usage data, in order to calibrate the UPS before real user were using it. Simulations are also the first step in UPS component evaluation process. After running the APOSDLE in the real world

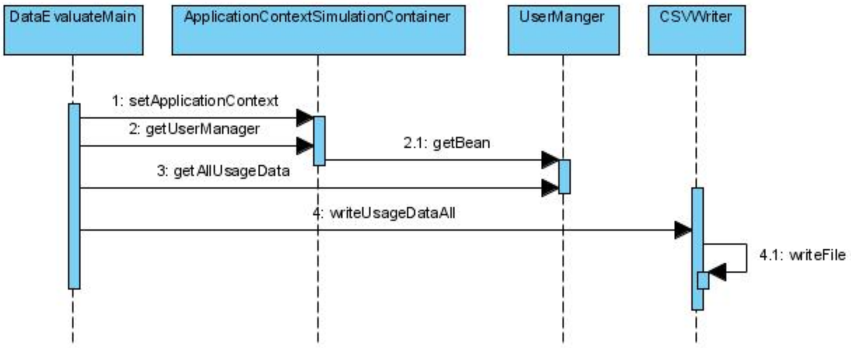


Figure 9.3: Evaluation component sequence diagram showing the workflow of the writing the usage data into .csv file

we should finally have the real data in the UPS data base. This data also needs to be evaluated. To conduct the evaluation with the real usage data there is a special component in UPS, namely Data Evaluation Component. This component uses a lot of infrastructure used by simulation framework. Same as simulation framework this component operates in standalone mode, meaning it doesn't need the whole APOSDLE system running to perform evaluations. Same as simulation framework only the UPS and depended components are needed. Basically the only new thing in this component was a special csv writer in charge in writing the results into .csv file. This .csv file will be imported into SPSS application for the visual presentation of the data. This is also very similar to the simulation framework, with the difference that the logging events are also considered. Beside KIE this type of events is particularly valuable to the evaluation purposes since they offer a better picture of the whole user activities in APOSDLE. For this reason it was needed to implement another writer especially designed for the evaluation component. To better understand the workflow of this component, see the sequence diagram (Figure 9.3)

9.4 Conclusion

The creation of the simulation framework was not only significant for the simulation purposes but also for the future service extraction. Here is showed how this can be done. Although the goal was not to create autonomous service, the creation of the simulation framework had a autonomous service as a side effect. Such services can be later used in different applications. The simulation framework it self was developed in the spirit of design patterns and therefore incorporating the best development practices.

10 Outlook and Future Work

This thesis describes the UPS Prototype 3 from configurability point of view and the simulation framework as a part of evaluation of the UPS component. As we saw the UPS Prototype 3 was completely re-implemented from scratch. The reason for not using the legacy UPS Prototype 2 was that the Prototype 3 used a new concept in the background, namely KIE. KIE concept opened a new perspective for the UPS. Suddenly it was possibly to make a assumption about the user on the really fine granular level. Each user action has become meaningful and the source of information about the user knowledge. At the same time we stand for a problem of interpreting this information. Apparently it was very hard to determine to which extend a certain user actions were relevant for the inferring purposes and how they should be interpreted. Another issue was if they have the same meaning throughout the different domains. Suddenly we face the fact of not knowing these things in advance. Therefore we had to introduce the new approach in developing the UPS namely configurability approach. This means having a number of different types of inference agents (user profile component configurations) prepared to work under different circumstances. These circumstances can be either new system, domains, different states within the same system or domain, working or behavior patterns etc. If for any reason the current configuration is not sufficient we should be able to easily change it with another configuration which is more suitable for the given circumstances. The obvious question at this point would be: how do we know which configurations are best equipped to work under certain circumstances? Well, we have to have some kind of proving mechanism to answer this question. The proving mechanism this theses presents is a simulation framework specially developed for this purposes. This framework tries to provide the system with real usage data.

After applying the configurability approach, we conducted a series of simulations (benchmarking between the different configurations) to narrow down the number of possible configurations in the system. These configurations were the candidates for the final version of the APOSDLE. The results of these simulations showed that we have to consider the knowledge aging factor when implementing the inferring algorithms. Aging factor in this case means the number of user actions used for the inferring purposes. Not just that simulation showed that such algorithms are having better performance but also provided the number of the actions which should be taken into account. At the end we were able to narrow down the number of the possible configurations from original six to just two.

The UPS was conceptually always considered as an autonomous service. This was not the case from technical point of view. The Simulation Framework (see chapter 9) showed that this is possible. The way how this was done was most probably not the optimal way but had showed that there are technical means of doing it.

If considered as a autonomous service with the different configurations possible, the application field is practically unlimited. There are two possible scenarios how this service can be further developed.

The first scenario tries to identify the user behavior types within one system and to apply the individual user profiling for each of these types. This presumes the extensive research in the area of the user behavior and usability of the intelligent adaptive systems. As a result we could have the adaptive approach on the user behavior level. For example there are users which tend to use one system on different ways although they may have the same knowledge level about the given topic. If the system is not able to differentiate between user types it is likely that the system will have different inferring results for these users. This may led to different and false outcomes in adaptation process.

The second scenario. We can imagine that there is a system which needs user profiling. If there is a proper underlying model representation of the domain there are no technical obstacles in using the UPS service for the user profiling purposes. We can go even further in our contemplation. Why not having a real web service running in the Internet and providing different distributed systems with a homogeneous user profiling capabilities. Today we have numerous social networking services (facebook, myspace, tagged etc.) or online shopping systems (amazon, eBay) or many Google applications which can provide such web service with the valuable data for the user profiling. In return all these distributed systems can exploit the results of this user profiling web service for their own purposes. Each one of these distributed systems can have his own (or more of his own) configuration defining the way how the UPS should work. The synergy created by using such UPS with distributed heterogeneous systems is enormous.

No matter which of these scenarios to follow it is obvious that there is a lot of potential for the configurability approach of the UPS.

List of Figures

Bibliography

[1] Jameson A. User modeling: An integrative overview. tutorial abis98:workshop on adaptivitiy and user modeling in interactive software systems, forwiss report. 1998.

[2] Jerry Banks. Introduction to simulation, proceedings of the 31st conference on winter simulation: Simulation—a bridge to the future. In *Winter Simulation Conference*, volume 1, pages 7–13, Phoenix, Arizona, United States, 1999.

[3] Peter Brusilovsky. Methods and techniques of adaptive hypermedia. In *User Modeling and User-Adapted Interaction*, volume 6, pages 87–129, 1996.

[4] Peter Brusilovsky. Adaptive hipermedia: An attempt to analyze and generalize. In *Conference on Multimedia, Hypermedia and Virtual Reality*, pages 288–304. Springer-Verlag, 1996a.

[5] Peter Brusilovsky. Adaptive and inteligent technologies for web-based education. kÃŒnstliche intelligenz,. pages 19–25, 1999.

[6] Peter Brusilovsky. Adaptive hypermedia. *User Modeling and User-Adapted Interaction*, 11(1):87–110, 2001.

[7] Peter Brusilovsky. Knowledgetree: a distributed architecture for adaptive e-learning. proceedings of the 13th international world wide web conference on alternate track papers and posters. In *International World Wide Web Conference*, New York, NY, USA, 2004. ACM New York, NY, USA.

[8] Peter Brusilovsky and Eva MillÃ¡n. User models for adaptive hypermedia and adaptive educational systems. In *The Adaptive Web*, pages 3–53. Springer-Verlag Berlin Heidelberg, 2007.

[9] Peter Brusilovsky, Michael Yudelson, and Vladimir Zadorozhny. A user modeling server for contemporary adaptive hypermedia: An evaluation of the push approach to evidence propagation. In *User Modeling 2007*, volume Volume 4511/2009, pages 27–36. Springer Berlin / Heidelberg, 2007.

[10] Peter Schwarz E. Weber G. Brusilovsky. Elm-art: An inteligent tutoring system on world wide web. In Springer Verlag, editor, *Third International Conference on Intelligent Tutoring Systems ITS-96, LNCS 1086*, pages 261–269, 1996.

[11] Andrea Bunt and Cristina Conati. Probabilistic student modelling to improve exploratory behaviour. *User Modeling and User-Adapted Interaction*, 13(3):269–309, 2003.

[12] Robin Burke. Knowledge-based recommender systems. *Encyclopedia of Library and Information Systems*, 69(32), 2000.

[13] R.R. Burton. Diagnosing bugs in a simple procedural skill. 1982.

[14] Brajnik G. C. and Tasso. A shell for developing non-monotonic user modeling systems. *International Journal of Human-Computer Studies*, 40 Nr.1:31–62, 1994.

[15] Cassandras C. *Discrete Event Systems.* Asken Associates, Boston, 1993.

[16] Hinrich SchÃŒtze Christopher D. Manning, Prebhakar Raghvan. Introduction to information retrieval. In *Introduction to information retrieval*, pages 116–117. Cambridge University Press, 2008.

[17] Cristina Conati, Abigail Gertner, and Kurt VanLehn. Using bayesian networks to manage uncertainty in student modeling. *User Modeling and User-Adapted Interaction*, 12(4):371–417, 2002.

[18] Knapell P. Arangno D. *Simulation Validation: A Confidence Assessment Methodology.* IEEE Press, Los Alamitos, 1993.

[19] Alberto DÃaz and Pablo GervÃ¡s. Adaptive user modeling for personalization of web contents. In *Adaptive Hypermedia and Adaptive Web-Based Systems*, pages 65–74. 2004.

[20] B. Berden B. De Lange B. Rousseau T. Santic S. Smiths N. Stash De Bra P., A. Aerts. Aha! the daptive hypermedia architecture". In *ACM Conference on Hypertext and Hypermedia,*, pages 81–84, Nottingham, UK, 2003. ACM.

[21] D. Smits De Bra P., N. Stash. Creating adaptive application with aha! pages 1–29, 2004.

[22] Nora Parcus De Koch. *Software Engineering for Adaptive Hypermedia Systems.* PhD thesis, Faculty of Mathematics and Computer Science, Ludwig-Maximilians-University Munich, 2000.

[23] Ronald Denaux, Lora Aroyo, and Vania Dimitrova. An approach for ontology-based elicitation of user models to enable personalization on the semantic web. In *Special interest tracks and posters of the 14th international conference on World Wide Web*, pages 1170–1171, Chiba, Japan, 2005. ACM.

[24] D.E. Egan. *Individual differences in Human-Computer Interaction.* Elsevier Science Publishers B.V., Amsterdam, 1988.

[25] Weber Gerhard and Peter Brusilovsky. Elm-art: An adaptive versatile system for web-based instruction. *International Journal of Artificial Intelligence in Education*, pages 351–384, 2001.

[26] Nicola Henze. Vorlesung aus personalisierung und benutzermodellierung, 2008. http://www.kbs.uni-hannover.de/Lehre/pers08/index.xml, last visited at 2008-09-29.

[27] Orwant J. *Doppelgaenger goes to school: machine learning for user modeling.* PhD thesis, MIT, 1993.

[28] Orwant J. Heterogeneous learning in the doppelgaenger user modeling system. *User Modeling and User-Adapted Interaction*, 4:107–130, 1995.

[29] F. Jennings and D.R. Benyon. Database systems: Different interfaces for different users. 1992.

[30] Nielsen Thomas D. Jensen, Finn V. *Bayesian Networks and Decision Graphs*, volume 2nd of *Information Science and Statistics.* Springer New York, 2007.

[31] Sleeman J.S.Brown. Inteligent tutoring systems. 1981.

[32] Blank K. *Benutzermodelierung fÃŒr Adaptive Interaktive Systeme: Architektur, Methoden, Werkzeuge and Anwendungen.* PhD thesis, University of Stuttgart, Germany, 1996.

[33] R. Kass and T. Finin. The role of user models in co-operative interactive systems. *International Journal of Intelligent Sytems*, 4(1), 1989.

[34] Alfred Kobsa. User modeling: Recent work, prospects and hazards; in adaptive user interfaces: Priciples and practise; practise; schneider-hufschmidt, m., kÃŒhme, t. malinowski,, 1993.

[35] Alfred Kobsa. Generic user modeling systems. *User Modeling and User-Adapted Interaction*, 11(1):49–63, 2001.

[36] Alfred Kobsa. Generic user modeling system. *User Modeling and User-Adapted Interaction*, pages 51–52, 2004.

[37] Alfred Kobsa. Generic user modeling systems. In *The Adaptive Web*, pages 136–154. 2007.

[38] Alfred Kobsa and Pohl W. The user modeling system bgp-ms. *User Modeling and User-Adapted Interaction*, 4(Volume 4, Number 2 / June, 1994):59–106, 1995.

[39] Alfred Kobsa, JÃŒrgen Koenemann, and Wolfgang Pohl. Personalised hypermedia presentation techniques for improving online customer relationships. *The Knowledge Engineering Review*, 16(2):111–155, 2001.

[40] Jaroslav Kuruc MÃ¡ria BielikovÃ¡ and Anton Andrejko. Learning programming with adaptive web-based hypermedia system aha! 2005.

[41] David Benyon Murray and Dianne. Adaptive systems: from intelligent tutoring to autonomous agents. pages 10–13, 1993.

[42] D.M Murray. Modeling for adaptivity. In *Proceedings of 8th Interdisciplinary Workshop, Informatics and Psychology, Scharding, Austria*, 1989.

[43] Fishwick P. *Simulation Model Design and Execution.* Prentice Hall, Englewood Cliffs, NJ, 1995.

[44] Joseph C Pasquale. Decaying confidence function for aging knowledge in expert systems. Technical report, University of California at Berkeley Berkeley, CA, USA, 1987.

[45] Michael Pazzani and Daniel Billsus. Content-based recommendation systems. In *The Adaptive Web*, pages 325–341. 2007.

[46] Nance R. *"A History of Discrete Event Simulation Programming Languages". The History of Programming Languages - II.* Association of Computing Machinery, New York, 1996.

[47] Elaine Rich. User modeling via stereotypes. *Cognitive Science*, 3(4):329–354, 1979.

[48] Elaine Rich. Users are individuals: individualising user models. *I nternational Journal of Man Machine Studies*, 18, 1983.

[49] Ana Paiva Self and John. Tagus- a user and learner modeling workbench. *User Modeling and User-Adapted Interaction*, 1995.

[50] John Self. Formal approaches to student modeling. Technical report ai-59, Lancaster University, England, 1991.

[51] John Self, G. I. McCalla, and Jim Greer. Formal approaches to student modelling. In *Student Modelling: the key to individualized knowledge-based instruction*, pages 295–352. Springer-Verlag Berlin Heidelberg, 1994.

[52] Roger D. Smith. Simulation article in "encyclopedia of computer science" http://www.modelbenders.com/encyclopedia/encyclopedia.html, 1998.

[53] Markus Strohmaier, Mathias Lux, Michael Granitzer, Peter Scheir, Sotirios Liaskos, and Eric Yu. How do users express goals on the web? - an exploration of intentional structures in web search. In *Web Information Systems Engineering*, pages 67–78. 2007.

[54] Schriber T. *An Introduction to Simulation Using GPSS/H.* John Wiley, New York, 1991.

[55] G.C. van der Veer. *Human-Computer Interaction. Learning, individual differences and design recommendations.* Alblasserdam, 1990.

[56] J. Vassileva. A classificaition and synthesis of student modeling techniques in intelligent computer-assisted instruction. In *ICCAL'90 Computer Assisted Learning*, pages 202–203. Springer-Verlag, 1990.

[57] Law A. Kelton W. *Simulation Modeling and Analysis.* McGraw Hill, New York, 1991.