INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA INFORMÁTICA




Scalable Intelligence for Scheduling Systems

BRUNO MIGUEL ALMEIDA CUNHA
Outubro de 2015

POLITÉCNICO
DO PORTO

Scalable Intelligence for Scheduling Systems

Bruno Miguel Almeida Cunha

Dissertation to obtain the Master of Science degree in
Computer Science, Specialization in
Knowledge-based and Decision Support Technologies

Supervisor: Ana Maria Madureira
Co-Supervisor: João Paulo Pereira

Porto, October 2015

Resumo

A personalização é um aspeto chave de uma interação homem-computador efetiva. Numa era em que existe uma abundância de informação e tantas pessoas a interagir com ela, de muitas maneiras, a capacidade de se ajustar aos seus utilizadores é crucial para qualquer sistema moderno. A criação de sistemas adaptáveis é um domínio bastante complexo que necessita de métodos muito específicos para ter sucesso. No entanto, nos dias de hoje ainda não existe um modelo ou arquitetura padrão para usar nos sistemas adaptativos modernos. A principal motivação desta tese é a proposta de uma arquitetura para modelação do utilizador que seja capaz de incorporar diferentes módulos necessários para criar um sistema com inteligência escalável com técnicas de modelação. Os módulos cooperam de forma a analisar os utilizadores e caracterizar o seu comportamento, usando essa informação para fornecer uma experiência de sistema customizada que irá aumentar não só a usabilidade do sistema mas também a produtividade e conhecimento do utilizador.

A arquitetura proposta é constituída por três componentes: uma unidade de informação do utilizador, uma estrutura matemática capaz de classificar os utilizadores e a técnica a usar quando se adapta o conteúdo. A unidade de informação do utilizador é responsável por conhecer os vários tipos de indivíduos que podem usar o sistema, por capturar cada detalhe de interações relevantes entre si e os seus utilizadores e também contém a base de dados que guarda essa informação. A estrutura matemática é o classificador de utilizadores, e tem como tarefa a sua análise e classificação num de três perfis: iniciado, intermédio ou avançado. Tanto as redes de Bayes como as neuronais são utilizadas, e uma explicação de como as preparar e treinar para lidar com a informação do utilizador é apresentada. Com o perfil do utilizador definido torna-se necessária uma técnica para adaptar o conteúdo do sistema. Nesta proposta, uma abordagem de iniciativa mista é apresentada tendo como base a liberdade de tanto o utilizador como o sistema controlarem a comunicação entre si.

A arquitetura proposta foi desenvolvida como parte integrante do projeto ADSyS - um sistema de escalonamento dinâmico - utilizado para resolver problemas de escalonamento sujeitos a eventos dinâmicos. Possui uma complexidade elevada mesmo para utilizadores frequentes, daí a necessidade de adaptar o seu conteúdo de forma a aumentar a sua usabilidade.

Com o objetivo de avaliar as contribuições deste trabalho, um estudo computacional acerca do reconhecimento dos utilizadores foi desenvolvido, tendo por base duas sessões de avaliação de usabilidade com grupos de utilizadores distintos. Foi possível concluir acerca dos benefícios na utilização de técnicas de modelação do utilizador com a arquitetura proposta.

Palavras-chave: Modelação do utilizador, Interação homem-máquina, Aprendizagem automática, Sistemas de escalonamento, Design centrado no utilizador, Inteligência escalável

Abstract

Personalization is a key aspect of effective Human-Computer Interaction. The ability to adjust itself to its users is crucial to any modern system, in an era where there is so much information and so many people interacting in so many ways. The creation of adaptable systems is a complex domain that requires very specific methods in order to be successful. However, still today there is no standard model or architecture to use on a modern adaptive system. The main motivation of this dissertation is to propose an architecture for user modelling that is able to incorporate separate modules required to create a scalable intelligence system with user modelling techniques. The modules cooperate in order to analyse users and characterize their behaviour, using that information to provide a customized system experience that will increase not only the usability of the system but also the user's productivity and knowledge.

The proposed architecture is composed by three components: a user information unit, a mathematical structure able to classify users and the technique to use when adapting content. The user information unit is responsible for knowing the several types of individuals that can use the system, for capturing every part of relevant interaction between itself and its users and also contains the database which stores that information. The mathematical structure is the user classifier and is in charge of analysing the users and classifying them into one of three roles: beginner, intermediate or expert. Both Bayesian and Artificial Neural Networks are used, and an explanation on how to prepare and train them to deal with user information is provided. With the user role defined, a proper technique to adapt system's content is required. In this work, a Mixed-Initiative approach is detailed which is based on allowing both the user and the system to gain control in the communication between them.

The proposed architecture was developed as part of the ADSyS project. ADSyS is a Dynamic Scheduling system to solve scheduling problems subject to dynamic events. It has a high complexity even for frequent users, hence the need for the adaptation of its content to increase its usability.

In order to evaluate the contribution of this work, a computational study of the user recognition was developed, as well as two usability evaluation sessions with distinct users. It was possible to conclude about the benefits of employing user modelling techniques with the proposed architecture.

Keywords: User Modelling, Human-Computer Interaction, Machine Learning, Scheduling Systems, User-centered Design, Scalable Intelligence

Acknowledgements

First of all, I would like to express my deep gratitude to my supervisor, Dr. Ana Madureira, for the given opportunities and being a crucial element of the creation and development process of this work. I would also like to thank my co-supervisor, Dr. João Paulo Pereira, for all the availability, opinions and constructive criticism.

A special thanks goes to my parents, Joaquim Cunha and Paula Cunha, for the encouragement to follow my dreams and to Rita Cardoso, for all the love and unconditional support.

At last, I would like to thank to my family, my friends and everyone that has supported me in any way during the realization of this work, contributing to its success.

Table of Contents

List of Figures

List of Tables

Acronyms and Nomenclature

Acronyms List

ADSyS	*Adaptive Decision Support System for Interactive Scheduling with Metacognition and User Modeling Experience*
AH	*Adaptive Hypermedia*
AHA!	*Adaptive Hypermedia Architecture*
AHAM	*Adaptive Hypermedia Application Model - First reference model in AH*
AM	*Adaptation Model*
ANN	*Artificial Neural Network*
BN	*Bayesian Network*
CBR	Case-based Reasoning
CPD	*Conditional probability distribution*
DM	*Domain Model*
IM	*Integration Mechanism*
ML	*Machine Learning*
UM	*User Model*
UML	*Unified modelling language*
XML	*eXtensible Markup Language*

1 Introduction

Regarding the Human-Computer Interaction field, personalization is a key topic (Shneiderman, 1998). Being able to adapt to its users is crucial to any modern system, in an era where there is so much information and so many people interacting in so many ways (Kay & McCalla, 2012). However, the creation of systems that cognize its users is a complex area, as their expertise is continuously evolving, increasing the complexity. This creates the need for new methods to develop new decision support systems that do not limit the user's choice but instead offer ideas, consent changes and are able to learn from the interaction with the user. The user's trust will increase bit by bit until the agent is reckoned as fully trustable. This approach is branded as Scalable Intelligence (Ramos, 2001), and is achieved by employing user modelling techniques that allow the system to known its users.

Reducing the time required by a user to obtain the desired result, either via suggestions or by providing the required tools earlier, can be a difference maker; that is the motivation for user modelling: to increase system's usability. A higher usability directly translates into higher effectiveness rates, requires less time to accomplish specific tasks (more efficient to use), is easier to learn and understand and creates a gratification feeling on its users, making them more adept of reusing the system.

In order to adapt any system to its users, a proper architecture should be planned. In this work, a user modelling architecture is presented. It is composed by a user information unit, a mathematical structure able to classify users and the technique to use when adapting content. The user information unit is responsible for capturing every part of relevant interaction between itself and its users, and also contains the database which stores that information (which is combined into the user model). In order to create the user information unit, it is necessary to perform an in-depth study on the current and potential system users. The mathematical structure is the user classifier. It is responsible for analysing user behaviour and allocating them into one of three roles: beginner, intermediate or expert. With the user role defined, a proper technique to adapt system's content is required. This technique should allow both the user and the system to gain control in the communication between them (e.g. the system can infer advantages and interact with the user and the user can directly ask for suggestions).

This document and the related work was contained in the dissertation of the 2nd year of the Master's Degree in Informatics Engineering of the Department of Informatics, ISEP. This work

was also done in the context of the R&D ADSyS (Adaptive Decision Support System for Interactive Scheduling with Metacognition and User Modeling Experience) project, supported by FEDER Funds through the "Programa Operacional Factores de Competitividade - COMPETE" program and by National Funds through FCT "Fundação para a Ciência e a Tecnologia" under the project: PEst-OE/EEI/UI0760/2014. ADSyS is a Dynamic Scheduling system applied to solve scheduling problems subject to dynamic events. It has a high complexity even for regular users, hence the need for the adaptation of its content to increase its usability. This project was developed at the research group GECAD (Knowledge Engineering and Decision Support research Centre).

1.1 Main goals and contributions

Despite the huge developments in user modelling, one of the main challenges remains finding a common approach for integrating user profiles (that support different users) within individual applications.

In this thesis, the main goal is to present a global architecture that is able to characterise its users, adapting its content to their level of expertise and knowledge about the system. It intends to be a guide on how scalable intelligence is implemented into ADSyS and how it can be applied in other contexts, serving as a starting point for future adapting systems.

Throughout this thesis a literature review was performed on multiple topics, such as:

- Establishment of adaptive hypermedia in order to understand the naissance of the user modelling concept;
- How to adapt any content to the user knowledge level, including a concise analysis of the domain and user models and solutions for content related questions (what to adapt, to what, why, where, when and how);
- History of user modelling as a research field and a greater overview of the user model;
- User modelling techniques, such as Bayesian and Neural Networks, and how to apply them to recognize users' knowledge;
- Succinct review on overall relevant topics, such as machine learning, Multi-Agent Systems and mixed initiative interaction;
- Current and future trends of the previous topics, such as Data Mining (in adaptive hypermedia) and virtual reality (user modelling).

The main contributions of this dissertation are:

- Proposal of an architecture able to incorporate each component required to create a legitimate scalable intelligence system with user modelling techniques. This includes the description on how to characterize current and potential system users, how to use that information, how to classify users by means of a mathematical structure and how to adapt system's content to the user's level of expertise;
- How to conduct usability evaluation sessions to assess the state of a Scalable Intelligence system (and how to prepare the system for them);
- Examination and demonstration of results from the developed work;

- Computational study on the ADSyS system to discover the differences of using a Bayesian Network and an Artificial Neural Network to classify users.

1.2 Document structure

This section describes succinctly each of the seven chapters that compose this dissertation.

Chapter 1 describes the developed work and its context, including the motivation, main goals and contributions.

Chapter 2 presents a literature review on the adaptive hypermedia research field, including a description on how to adapt content to the users' knowledge.

Chapter 3 explains the user modelling concept and indicates the available techniques to classify users according to their knowledge level.

Chapter 4 describes the ADSyS system architecture comprehensively in order to understand the decisions made and the proposed user modelling architecture.

Chapter 5 depicts the main objective of this dissertation, which is the proposal of an architecture for user modelling. It includes the detailed process of the research and creation for each component.

Chapter 6 displays the performed experiments and the results obtained in order to validate the developed work, incorporating a computational study on the matter of which network to use to classify users (Bayesian or Artificial Neural Network) and the conducted usability evaluation sessions.

At last, **Chapter 7** presents the conclusions learnt and examined with this work and the limitations of the developed architecture. Future works related to this dissertation are also presented.

2 Adaptive Hypermedia

This dissertation focuses on the study and creation of a user modelling module to be incorporated into a high complexity system, ADSyS. User modelling, as a research area, has emerged from the Adaptive Hypermedia (AH) field.

This chapter describes how to use an AH approach to increase the utility of a system. It starts by analysing AH in an historical perspective on the first definition and systems using it, until 1997. After that, an evolution since the big development year in AH (1998) until the current era is presented, including a presentation of reference models and systems. Then, a proper study on how to adapt content on any system is stated, including an description of the six fundamental AH questions, introduced by Brusilovsky (1996). Section 2.3 also contains a description of the components needed to create an AH system, including the description of how the user model fits and operates with other components. The chapter ends by presenting the latest developments in the AH field, including approaches such as ontologies or group adaptation.

2.1 Creation and first works

AH is a growing research topic where hypermedia (content as images or videos) is adapted in accordance to a UM. The traditional approach consists in offering a set of static pages and hyperlinks (hypermedia references) to every user. However, if the target system has a disperse user population, that approach is unable to satisfy all user needs (Brusilovsky, 2000). The focus of AH researchers is to develop methods, techniques and architectures that are able to adapt a systems' content to the needs and preferences of its users. In AH, what the users see is suited to their objectives, preferences and knowledge model (Brusilovsky, 2001). For instance, in an education system with static hyperlinks each student will receive the same suggestions and explanations, while in an AH system those interactions are customized in conformity with students' needs and difficulties. Another non-educative case is an AH virtual museum, that will adapt previously visited objects in accordance to the user trajectory, as presented in Oberlander & O'Donnell (1998).

In Eklund & Brusilovsky (1998), an analysis of the scientific development in AH is presented, an area that appears in the beginning of the 1990 decade in the field of educational systems – that would be later established as e-Learning systems or Technology-Enhanced Learning. These systems appeared first on education due to the "lost in hyperspace" problem (Edwards & Hardman, 1989), where students did not know how to use the information provided by the system. Given the maturity of the User Modelling and Hypertext research, the crossing of ideas between these two areas was natural. Several teams recognized the static Hypertext problem on different applications (Eklund & Brusilovsky, 1998) and started an exploration to adapt its behaviour, creating AH. The first system to adopt AH was Manual Excel (de La Passardiere & Dufresne, 1992), and its focus was in developing historic mechanisms – as the Three-stage footprint (not seen, partially seen and seen) – and guidelines that adapted to the student, based on a calculus that included the number of errors and tries. In de La Passardiere & Dufresne (1992), the authors performed some tests using a system with and without adaptation, comparing the usage of students in an environment deprived of adaptation to a group of students that operated with the full system, including custom advices and historical mechanisms. Right from the beginning the authors concluded that both systems were very dissimilar, given that one offered feedback and adjustment, while the other stayed fully static during its use. The interpretation is that the importance of presenting an individual progression is as relevant as allowing the user to influence the information domain of the system being used, and that the effort to apply AH techniques is clearly beneficial.

Other studies and systems of reference are (Brusilovsky & Pesin, 1995; Weber & Specht, 1997). The first is contemplated, for its era, as the piece with the most empirical power to sustain the hyperlink adaptation method in AH systems. The authors of ISIS-Tutor, which used diversified symbols and colours to mark content from the current page related to other pages. Those indications were linked to states as "ready to learn", "in progress" or "not ready to learn" – following the three-stage footprint mechanism. In Brusilovsky & Pesin (1998), the authors present the final version and evaluation of Isis. The work suggests that learning while taking advantage of diversified marking and several adapted links is faster, more goal-oriented, and reduces significantly the number of required steps to archive an objective. Several results are described, pointing in particular to using links adaptation on information retrieval domain as the most efficient method. The authors also debate regarding the advantages of using adaptive or static layouts (favouring the adaptive) and about the use of other techniques (that would be real in future works), such as interactive samples, study cases or problem solving.

2.2 Current state

It was in the year of 1998, almost at the beginning of the new century, that the AH field gets more attention from the scientific community, experiencing a substantial growth (Popescu, 2010). In 1999 the first reference model on this field appears, branded as AHAM (Adaptive Hypermedia Application Model (De Bra et al., 1999)), and a very close adaptation of this model, identified as AHA! (Adaptive Hypermedia Architecture (De Bra et al., 2006)), was published for the scientific community use. This reference model united the AH researchers, providing a common architecture responsible for guiding the research activities in multiple directions.

In the last decade numerous systems that benefit from AH were developed, particularly in the e-Learning field, which was the main focus of AH research. The most relevant systems are KBS Hyperbook (Henze, 2000), APeLS (Conlan et al., 2006), the Interbook system (Brusilovsky et al., 1998) and WINDS (Specht et al., 2002), amongst others.

Some attempts were made in order to expand the AHAM reference model that can be found on scientific literature. The most significant proposed models are Munich and Goldsmith. The Munich reference model (Koch & Wirsing, 2002) tries to capture each of the system's architecture in a UML notation. The Goldsmith model (also known as GAHM) (Ohene-Djan & Fernandes, 2002) was later presented, in an attempt to combine itself with the other key models (AHAM and Munich). However, the combination outcome was unsuccessful due to not obtaining a unified conceptual representation. Nevertheless, the study lead to finding out problems with those reference models, specifically questions related to their implementations and issues with the meta-data employed by those systems (Ohene-Djan et al., 2003), which would be used to their respective improvement.

For the most part, the new proposed systems resulted in new terms, models, methods, concepts and prototypes. The previously described ideas were transferred to these new works, uncovering new utilities and application prospects. Nonetheless, in spite of the variety of proposed AH systems, there is not a consensus regarding the correct approach, which architecture is the best to apply in modern, contemporary AH systems.

2.3 How to adapt content

The implemented adaptation to a given system is defined by answering to six fundamental questions, introduced in Brusilovsky (1996):

- What can we adapt?
- To what can we adapt to?
- Why do we need adaptation?
- Where can we apply adaptation?
- When can we apply adaptation?
- How do we adapt?

In Brusilovsky (1996), it is also proposed a scheme (Figure 1) that structures the proposed questions, ordering them. By answering them, the description of the adaptation process is created. This process is, in most of the cases, started by the user when establishing the goal of the adaptation – solving the "why?" question. "What?" and "to what?" are related to the domain and user models, respectively. The following queries are the "when?" and "where?" which support the definition of the adaptation context. At last, "how?" inquiries regarding which techniques and methods to use when adapting the system, both on a concept and implementation level. Every single answer is then brought together, resulting in a detailed definition of the AH system.

The first models for AH systems (described in the previous sections) stated that the adaptation of a specific system relies on three factors: the Domain model (DM), the User model (UM) and the Adaptation model (AM):

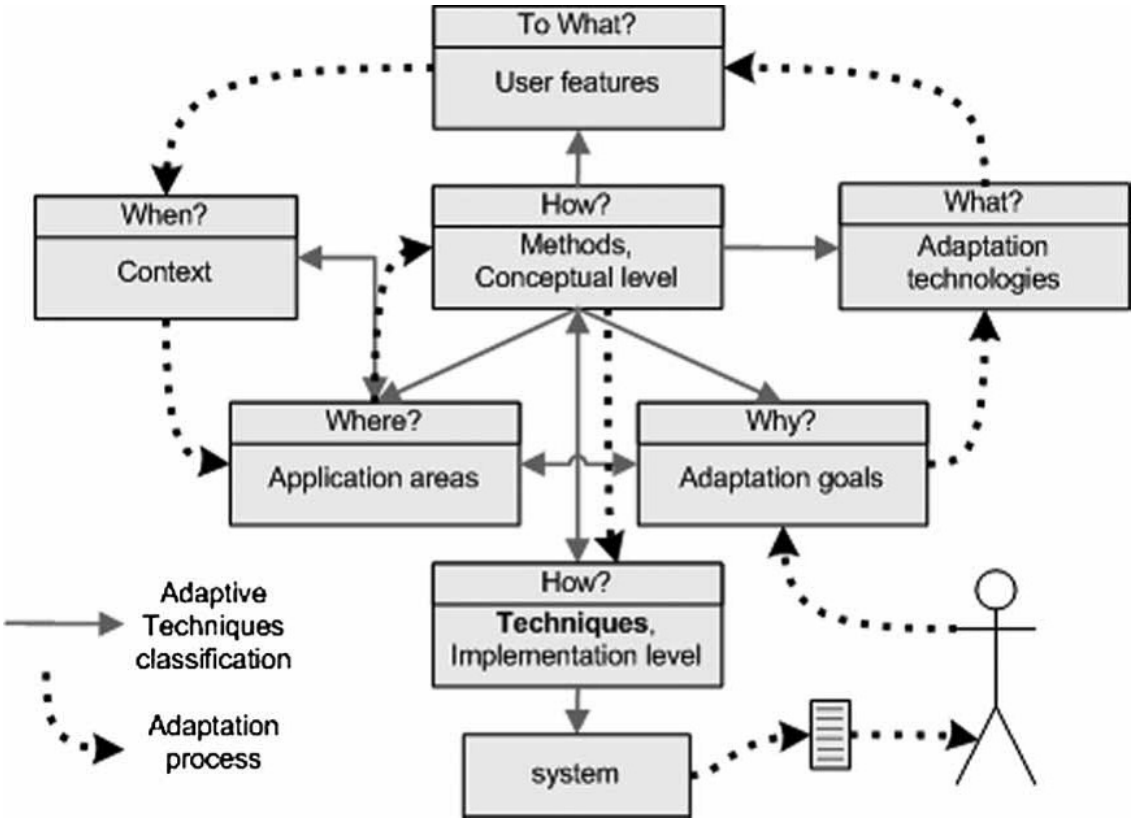

Figure 1 – Adaptation process of AH systems: structural questions (Knutov et al., 2009)

I. The adaptation should be based on a DM that describes the domain conceptual representation and its structure. This model indicates the relations between concepts and how they are connected in terms of information (such as fragments, pages or information units (Henze, 2000)). The DM provides the answer to "what can we adapt", describing the domain structure and the information that needs to be customised, conceiving the link between concepts and their respective representation (Knutov et al., 2009).

II. The UM is created with the purpose of representing the user information: objectives, preferences, interests, action history, user knowledge and several other facts that may be useful for the customization (Knutov et al., 2009). The UM answers "to what can we adapt" (Brusilovsky, 1996): what are the aspects of the users' interaction with the system that can be taken into account when adapting, and which functionalities can the system adapt. The UM may also satisfy the "why" Question, providing information about user's goals.

III. The goal of an AH system is to shape its navigation structure, presentation and content to the knowledge level of its users, their goals and navigation model, amongst other (Knutov et al., 2009). The AM is then required, indicating how the relational concepts of the DM influence the navigation and its properties (e.g. if the system should guide the user towards information about a specific domain (De Bra et al., 1999)). This model can also solve the "when" and "where" queries, and might bring the "what can we adapt" again, with the interpretation of the constraints in the structure of DM relations.

This division in DM, UM and AM emphasizes the separation between the big question marks in AH systems. Then again, this partition still mixes some questions (it only has three elements for six questions), and a more thoughtful specialization of the layers is needed (Knutov et al., 2009) in order to accomplish a better separation and offer the proper granularity in its architecture.

2.3.1 What? – The Domain Model

The DM of an AH system is composed of concepts and its associations (Knutov et al., 2009). A concept denotes abstract data from the system field. These concepts create a hierarchy, enabling that each one to be either an atomic concept or a composite concept (that has child concepts). As an example, in Interbook (Brusilovsky et al., 1998), the domain concepts are mapped into a document space which contains multiple items. This also happens in Henze (2000), where the system uses a knowledge foundation. AH systems gather concept fragments, using that information to apply adaptive techniques directly to a group of them, to accomplish the desired user adaptation and support. In the AHA! (De Bra et al., 2006) and AHAM (De Bra et al., 1999) models, the concepts representation consists in pages containing fragments, as illustrated in Figure 2.

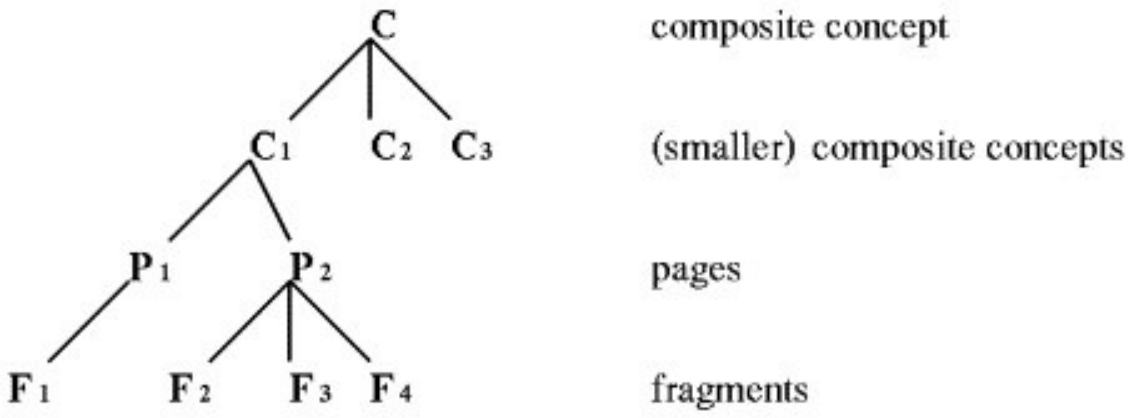

Figure 2 – Concept hierarchy in the AHAM (De Bra et al., 1999) and AHA! (De Bra et al., 2006) models

In general, the DM is considered as a structure similar to Figure 2, that might be created by a domain expert (Knutov et al., 2009). This is a fixed structure, and the system customization can only be provided within the limits of the knowledge modelled into the DM. However, an increasing effort has been developed recently to move away from this restriction into “open corpus adaptive systems”, as defined in Brusilovsky & Henze (2007) aiming to the operation of documents that are not known at design time and can be dynamic – constantly changing and expanding (Brusilovsky, 2008). Open Corpus adaptation is described in section 2.4.1.

2.3.2 To What? - The User Model

The adaptation in AH systems is made based on user characteristics, symbolized on the UM. However, the concrete method that each system uses to adapt may vary, and some AH systems even decide to use something else rather than user traits (Knutov et al., 2009). In Kobsa (2001) is presented the distinction between adjusting to user data, usage data and environment data.

The first is based on the adaptation goal, usage data relates to the user interaction, and the last contains information not associated to the UM or interaction with the system.

A typical UM consists of entities (Conlan et al., 2006) which are connected to a number of attribute-value pairs. Most entities in the UM represent concepts from the DM. In Brusilovsky (1996), user modelling is defined as working in a loop with adaptation, as portrayed in Figure 3. The relation between the DM and the UM is that the UM fits "on top" of the DM, mapping the users traits over the domain knowledge. This is achieved by associating attribute-values with each part of user knowledge. Some works that follow this approach are KBS Hyperbook (Henze, 2000) and APeLS (Conlan et al., 2006).

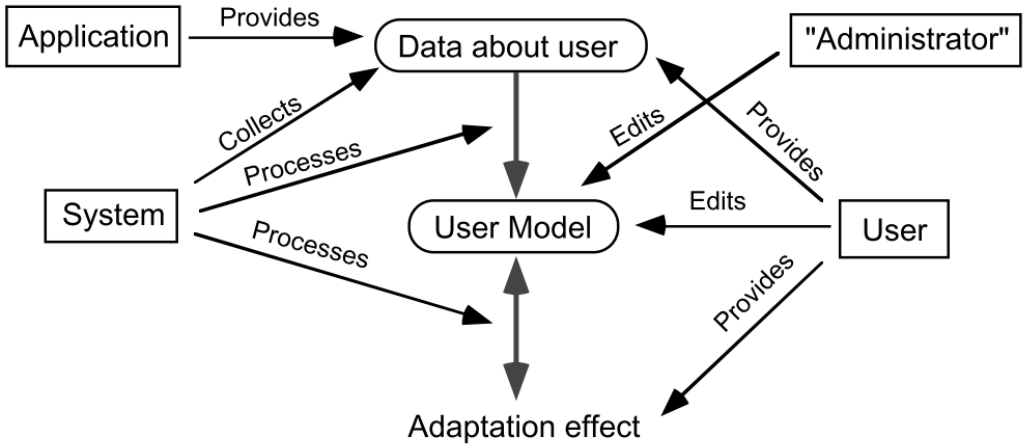

Figure 3 – User Modelling: adaptation loop in AH systems (Brusilovsky, 1996)

The UM is the motivation of this thesis. A proper study on UM, how to create and implant it into an AH system, collect and deduct user information, amongst other factors, is presented in chapter 3.

2.3.3 Why? – User goals

Although the user goal might be inferred from the UM, that is not the most natural and correct approach. When contemplating on goal-driven AH systems, the goals alone are not sufficient. A goal is not only an objective, but instead a structure that also includes tasks, requirements and workflows, describing a more procedural method. In contemporary AH systems, a goal model should be contemplated (Knutov et al., 2009).

Some proper examples of goal-driven adaptation are LAOS (Hendrix & Cristea, 2008) where the "Goals and Constraints model" is proposed, TANGOW (Martín et al., 2006), APeLS (Conlan et al., 2006) and KBS Hyperbook (Henze, 2000). Particularly on KBS Hyperbook, the authors mapped user defined tasks onto "project" units, with each one representing an index of "knowledge items". This method provided a detailed approach where "projects" implied real application issues that could be solved by a sequence of tasks, each interacting with a new concept. With this approach, it is possible to create a vast project collection to accomplish, simultaneously, multiple user goals.

A goal-driven approach consists of forming a structure of goals and its corresponding tasks, creating a workflow that needs to be followed to complete a requirement. This structure has to be aligned with a DM, in order to explain the mappings between both models, achieving an improved adaptation.

Deducting a goal from what the other users have been doing within the system is also possible. This inference uses the navigation patterns from other users to give recommendations to the current user (detailed in section 2.4.3). This fact introduces a new paradigm, where user goals are not restricted only to a list from where they could select their objectives. The new generation of AH systems has to support dynamic goal creation, either when it is projected by a domain expert or when the suggestion is originated from the system, as defined and exemplified in Mei & Easterbrook (2007).

2.3.4 When and Where? – Context

AH techniques can, in theory, be implemented in all types of systems. However, there are two main fields where it has been applied: e-Learning (AH primary focus) and, more recently, online information systems – that can range from TV guides (Tintarev & Masthoff, 2008) to item recommendation (Rutledge et al., 2008), or even social web (Farzan & Brusilovsky, 2006). The diversity of AH systems is increasing with time, being difficult to capture the whole frame of new developments and approaches (Knutov et al., 2009).

In recent AH systems, context issues started to have an important role. The popularity of context aware systems is undoubtedly increasing, but this type of development is incredibly field-dependent (as described in section 2.4.5). An example of a system that uses context-sensitive user interactions has been developed in Ardissono et al. (2008). Context awareness may replace the definition of the system's environment, allowing it to escape from a narrow field of application and providing flexibility that evolves with the context. An adaptive system should follow its dynamic context (Knutov et al., 2009).

The term "context" may be applied not only to the system where the adaptation is performed, but also to the environment where the system is used. The application context answers the "where" question (Brusilovsky, 1996); the environment context (e.g. time, network bandwidth or month) solves the interrogation of "when" to apply the adaptation.

2.3.5 How? – Adaptive methods and techniques

In Brusilovsky (1996), adaptive techniques and methods are defined as ways of delivering adaptation and their generalization, respectively. Techniques are a part of an AH implementation and are usually unique, using a particular approach or algorithm. Methods represent the simplification of an existing technique. Each method displays an idea of how to approach adaptation and can be implemented by several techniques. Techniques and methods can be applied simultaneously to page content, to its presentation or to adapt the system navigation, known as content adaptation, adaptive presentation and adaptive navigation techniques, respectively.

Merging methods and techniques with the information gathered from the previous "answers" creates an AH toolkit, pictured in Figure 4 (adapted from Knutov et al. (2009)).

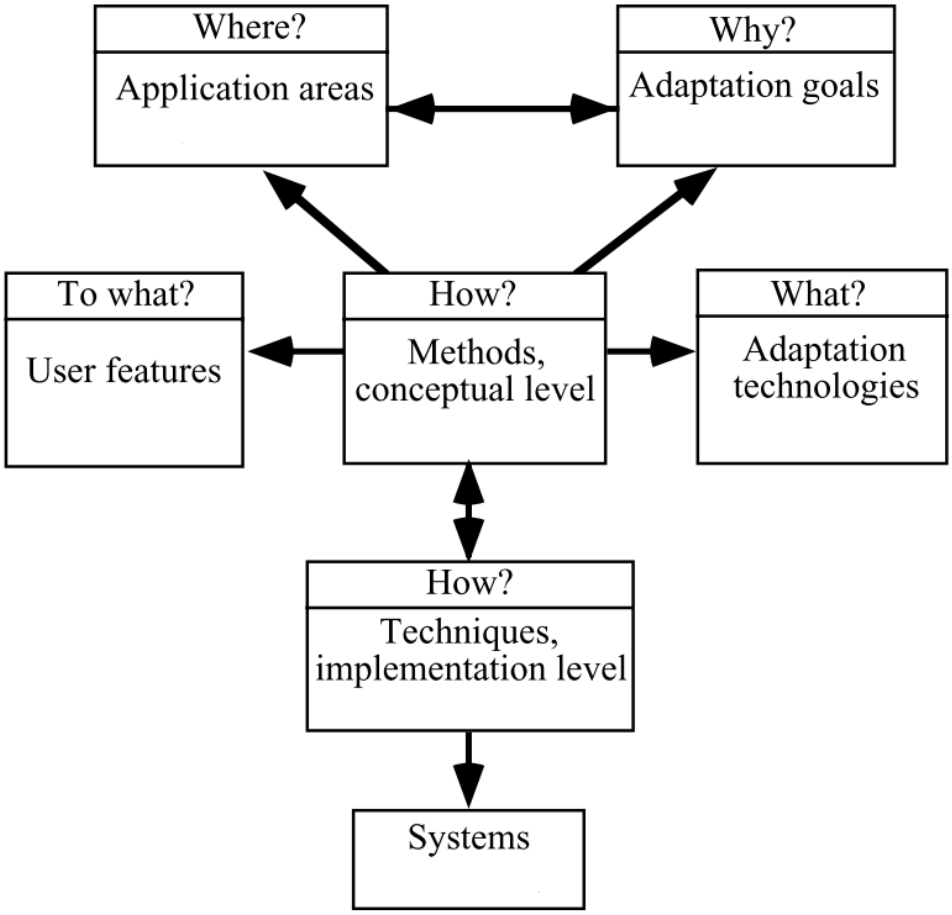

Figure 4 – AH toolkit. Arrow stands for a 1-to-N relationship

Pioneering AH systems – dedicated almost exclusively to the education field – controlled the adaptation decisions (e.g. what to show or which step to take) exclusively, and although some systems still impose their decision upon its users, AH development trends to offer users an increasing control (León et al., 2005). This justifies the need for techniques that adapt the system navigation (e.g. adaptive presentation) which do not change the information or available paths on the system, instead using variations on its appearance to insert recommendations to the user. The use of these techniques is influenced by the increasing knowledge that systems have about its users, and can create better adaptation results by using their characteristics (Challis, 2005).

Content Adaptation

There are, effectively, two ways to adapt information: either by drawing attention to a certain content or by showing/hiding specific information. The variance is whether the information is available or hidden. Some content adaptation techniques are sorting, zooming or scaling (Knutov et al., 2009). These methods might also be considered as presentation techniques, given that they do change the presentation.

Adaptive presentation

Some presentation techniques can be used to adapt the contents of a page to its user, as described in the previous section. Nonetheless, some other methods are applied for different reasons, usually to adapt the system layout according to user preferences. Layout adaptation might be needed for multiple reasons, such as device adaptation (e.g. devices with limited screen size) or to comply with a predefined format (De Bra et al., 2008).

Adaptive navigation

This technique supports a customized access to information on the system, by adaptively adjusting the appearance of shortcuts and links in a certain page (Brusilovsky, 2007). The evolution of these technologies allowed users not only to be more effective, but also to reduce their navigation overhead and increase their satisfaction (Kaplan et al., 1993; Brusilovsky & Pesin, 1998). Some noteworthy methods used to provide guidance to users are link ordering, hiding, annotation and generation (Brusilovsky, 2007); to produce these methods there are several mechanisms, which can be classified into different categories: simple adaptation, content-based, social and indexing-based mechanisms.

Simple adaptation techniques do not require advanced algorithms, and usually are history-based or trigger-based, as in de La Passardiere & Dufresne (1992) – one of the first AH systems. An additional category is content-based mechanisms, which make a decision on whether to suggest a link based on page content. They typically operate by iterating keyword vectors and comparing them with user interests, as described in (Pazzani et al., 1996; Armstrong et al., 1995). Social mechanisms are built on the sense of social navigation, which takes advantage of the natural tendency of people to follow directly or indirectly signals from the activities of others (Brusilovsky, 2007). A standard method to implement direct social navigation is to mark links to content that are currently being visited by other users, as accomplished in Kurhila et al. (2002). Social mechanisms are further described in 2.4.3. Nevertheless, the most powerful and popular methods to provide adaptive navigation support are indexing-based techniques. An indexing-based approach purpose is to represent information about each page that can be associated to the UM and use that association to make the decision on whether and how to provide guidance, similarly to content-based mechanisms. The difference comes from the representation, with indexing-based approaches using a concept-level representation (Brusilovsky, 2007), which is more precise. A successful system that uses index-based adaptation is presented in Brusilovsky et al. (1998).

Circumstantial adaptation possibilities

Depending on the type and content of a system, some other adaptive possibilities can be applied. Systems that operate with photographic and multimedia content have a specific category of adjustment: Adaptive multimedia presentation. It uses not only the textual techniques presented in other approaches, but also takes advantage of the pictorial information (such as width and height) available in images, videos and other types of multimedia resources. A comprehensive study on adaptive multimedia presentation is available in Hanisch et al. (2006).

AH systems, which contain not only content to be read, but also tools to interact with multiple resources, benefit from another adaptation approach: tools adaptation (Knutov et al., 2009). This form of adjustment is very specific and varies from system to system, but generally results in providing the user with different sets of features depending on their expertise. Systems that

use this approach are, amongst others, described in (Madureira et al., 2014b; Carro et al., 2003; Triantafillou et al., 2004).

2.4 Current trends in AH systems

Current AH systems have been evolving constantly, using new discoveries and technologies to provide a superior user adaptation. These developments are already established and have been used in some AH systems, but a unifying architecture is still lacking, that combines them together to create a complete, generic AH system.

2.4.1 Open corpus adaptation

A closed corpus AH system operates on a closed corpus of documents - where documents (description of a systems' content) and theirs connections are known at design time. An open corpus system operates on an open corpus of documents (e.g. a set of documents that are not known at design time and can dynamically change) (Brusilovsky & Henze, 2007). In order to perform adaptation to an unknown document or group of documents, the mapping between concepts and documents' content can only be done at run-time, bringing the fields of hypermedia, databases and information retrieval together. Open corpus is not a completely new area, but is yet to make an impact in AH systems, due to the high difficulty of implementing this technology - it is hard to foresee how external content, outside the control of authors and developers, should be adapted. Nonetheless, some developments have been made, by extending AH open corpus approaches to benefit from developments in other research areas, such as natural language processing (Levacher et al., 2012).

2.4.2 Ontologies

Several AH systems' authors are the creators not only of the system itself, but also of its content. To be able to combine the adaptation provided by different systems, the employed concepts and their meaning must be agreed beforehand. In this definition, instead of choosing an arbitrary structure, AH systems can integrate ontologies. The challenge now is to overcome the capacity of dealing only with a single ontology, creating the capability of reasoning over different ontologies (Aroyo et al., 2007).

2.4.3 Group Adaptation

A typical AH system executes its adaptation process for every user, with a different outcome based on the users' characteristics. However, this approach can be extended by considering actions executed by other users and the adaptation that was provided to them. Determining the right groups of users is a current focus in AH investigation, noted on some works that tried to incorporate group adaptation in their adaptation process (Carro et al., 2003). A recent work that claims to achieve this adaptation is detailed in Smits & De Bra (2011), but the authors restricted its use due to concerns with user privacy. The major issue in this method is the lack of a new reference model that associates group models with a single UM. Another issue is the absence of rules for an individual user that, triggered by user actions, generate updates not only

on both models (group and single user models), but also of the models influencing the adaptation performed for a user (e.g. the AM that affects users of the same group) (Knutov et al., 2009).

2.4.4 Data Mining

Group adaptation, described in the previous section, can create information that might be used to improve each type of adaptation in a system, when combined with Data Mining – e.g. Clustering users in groups based on their patterns and characteristics can be used to automatically generate suggestions, based on the group interests (Yan et al., 1996). As many other developments, Data Mining is introduced in AH by e-Learning systems (Romero et al., 2003), and its main benefit is the creation of a new, innovative method to determine the required adaptation.

2.4.5 Context awareness

This approach extends or replaces the AM by using context awareness (using a proper context model). As described in Knutov (2009), this method will help to decouple and make AH systems less integrated with and dependent upon the environment in which they are used. A proper study and accurate definitions of context and context awareness is available in Dey & Abowd (1999). It defines context as any information that can be used to characterize the situation of an entity (which can be a person, place or relevant object). A system is context aware if using its context to provide relevant information or services to the user.

Although the concept of using the context in AH being acceptable (as described in section 2.3.4), few works use a factual, accurate context awareness approach. A recent case study of a system using this method (even if using a proprietary model), is available in Motti (2013), with an uncertain degree of success.

2.5 Summary

AH is a research field that aims at improving users experience operating a system, adapting its content in order to improve the usability. AH appears in the beginning of the 1990 decade (related to e-Learning), and ever since as seen multiple improvements, including new terms, models, methods, concepts and prototypes.

In order to create an AH approach on a common system, 6 fundamental questions regarding the adaptation need to be answered: What to adapt, to what, why, where, when and how. These answers lead to the creation of 2 models (user and domain), to the discovery which techniques to use and to define appropriate times and places to adapt contents (further described in detail in section 2.3).

Multiple developments have been made in the last years, especially on using system's context to improve the adaption and in creating a dynamic modelling solution (open corpus adaptation, section 2.4.1). Nonetheless, and in spite of several improvements and contributions, still today there is no agreement on the "correct approach" – which standard model or architecture to use

on a modern AH system. The best practice remains to study the target system and apply the most appropriate methodology, according to the circumstances.

3 User Modelling

User modelling is a research field that investigates how users interact with technology and how to adapt system's content to all users and their necessities. In order to understand how to apply user modelling techniques to increase a system's usability it is required to understand, both in theory and practice, how they work and how to properly apply them.

This chapter presents substantial information on user modelling, including a concise history review (from the first user modelling system to modern ones), a deeper investigation on what is and how to construct a UM (able to represent the system's beliefs about its users) and modelling techniques that are used to build the model. The presented techniques vary from the simpler ones (such as stereotypes) to the most used (Bayesian or Neural Networks). Current trends in user modelling are also examined to give a glance at what the future might hold for this research field.

3.1 History

User modelling has become a central subject for anybody interested in understanding how users interact with technology. Its main objective is to customize and adapt systems to users' specific needs so that they can engage the system in their own terms, increasing its usability.

Pioneering user modelling systems had no distinction between specific modelling components and system components that performed other tasks (Kobsa, 2001). It was in 1979 that the first user modelling systems were presented (Cohen & Perrault, 1979; Rich, 1979a). Since then, multiple user modelling systems have been developed, with a growing concern to separate the modelling component in order to be reusable in future systems. In 1986 GUMS (General User Modelling System) was published (Finin & Orager, 1986), providing developers with a tool to define stereotype hierarchies and their respective members and rules (using Prolog facts). This approach was designed to work in parallel with the main system, accepting new facts (validating for inconsistencies) and answering queries from it. GUMS set the framework for future user-adaptive systems (Kobsa, 2001). External frameworks for user modelling (such as GUMS) were later defined as "user modelling shell systems", both by Kobsa (1990) and Finin (in an updated version of GUMS (Finin, 1989)). Recent developments are focused on incorporating current technologies to model users, such as ontologies (Sosnovsky & Dicheva, 2010; Heckmann et al., 2005), XML (Fernandes et al., 2014), UML (Heckmann & Krueger, 2003), Multi-Agent Systems

(Vassileva et al., 2003) and using a personas based approach (Brigham, 2013). Despite the huge developments in the research area, one of the main challenges remains the absence of a common approach for integrating user profiles that support different user models within individual implementations, and the migration of profiles from one implementation to another.

A recent concern is related to data privacy. Privacy is not considered a crucial aspect in the majority of current systems. There is the opinion (Mohamad & Kouroupetroglou, 2014) that privacy issues do not need to be assessed if the system uses stereotypes (and not specific user preferences). However, the use of such stereotypes can be seen as implicit hints about the specific user (e.g. their preferences or abilities/disabilities). Therefore, privacy (in terms of user modelling and data) is one of the main issues to be tackled, especially to propel the adoption of such systems from wider audiences.

Another relevant debate in user modelling is the type of modelling to be abided in the area: user characteristics or user preferences. The first allows the system to know about the characteristics of a person and other contextual information (such as the device used and other environmental variables), that can reveal what kind of adaptations are needed in order to make the product accessible to its users. The second keeps information about the users' preferences, that can be chosen by the users themselves (e.g. over a question answering session) or could be recorded on the background by keeping track of users' actions and reactions on various adaptations. Both have advantages and weaknesses (e.g. the first approach is better in dealing with contextual variables, the second has an advantage in being easier for users to train, providing a more accurate representation), so further research into this topic must be done (Mohamad & Kouroupetroglou, 2014).

3.2 The user model

A UM represents the system's beliefs about its users, providing the required information to adapt to their needs. As described in Froschl (2005), once the necessary information to be stored in the UM is decided, the next stage is to define where the data will be gathered. The construction of the UM is the first step (using an appropriate technique, as described in section 3.3), and then it has to be filled with the initial (or default) data. Then, the UM goal is to keep up to date with users in order to represent truthfully their current state.

3.2.1 Content

An accurate UM must contain data regarding the user's preferences, goals, interests, domain knowledge and progress. The information on a UM can be split into two categories: domain specific information and domain independent information.

Domain specific information characterizes the user's state on a particular subject (related to the DM). This information is combined into a model, defined by Brusilovsky as Knowledge model (Brusilovsky, 1994). There are several types of knowledge models: Scalar, Overlay, Error and Genetic (Froschl, 2005). Scalar models categorize user's knowledge on the domain by an identifier from a limited range (e.g. a number from 1 to 5), and are the simplest form of knowledge models. The overlay model describes the user comprehension on a subset of the DM, and uses a scalar measure (e.g. probabilities or flags) to estimate user comprehension on each topic (Henze & Nejdl, 2004). The Error model makes possible to describe user's erroneous

behaviours and their reasons. Genetic models help understanding the user's progress throughout the system (e.g. a user that starts with very specific knowledge and evolves into a more generalized one). Other types of information might be stored on a UM depending on the system's context and DM, such as the user prior knowledge about the domain, number of helps asked or time required to solve a problem.

Domain independent information includes facts such as user goals, cognitive abilities, experience, preferences, factual and historic data (Brusilovsky, 1994). User goals establish why the person is using the system. Cognitive aptitudes are abilities for differing types of cognitive performance (e.g. math or musical aptitude). User experience (or background) includes skills that might influence the system operation (e.g. profession or perspectives). Preferences are considered as not inducible by the system, and the user as to transmit them, directly or indirectly. Preferences can also be used to group users (as described in section 2.4.3). Factual and historical data contain information such as demographic data (e.g. name and age), and are used to initialize the UM.

3.2.2 Initialization

The initialization of a learner model represents the practise of gathering information about the user, moulding it into the UM. In the field of recommender systems this is a well-known problem, identified as "cold-start" (Schein et al., 2002): the system is not able to recommend something to a user that is not known yet. Self (1994)defines three methods to initialize a UM: explicit questions, initial testing or using stereotypes.

Explicit questions

In this method, the initial UM is filled by questioning the user. Although effective, the problem is not only finding the appropriate amount of questions, but also inferring the optimal information out of the answers, e.g. a high quantity of questions might be a nuisance (Tsiriga & Virvou, 2003). An interesting approach is presented in Kurhila et al. (2002), where an adaptive questionnaire is described, using Bayesian methods to optimize the inquiries.

Testing

If the user is prompted to take an initial test, the UM data can be inferred by analysing the results. The test should be well planned, in order to decrease its length. Some methods can be applied to achieve this, such as neighbourhood knowledge (Self, 1994): if elements A and B are in the same neighbourhood, knowing A implies knowing B.

Stereotypes

The UM may use a default state for each user that logs into the system. This is a powerful approach (requires no input from the user), but also the least accurate. The default UM is then refined by constantly observing the user throughout its use of the system (Han, 2001).

3.2.3 Updating

Since the user preferences and characteristics are not constant properties, a change over time has to be considered by the UM. The information used to update the UM can be retrieved from several sources. According to Kinshuk (1996), there are four methods to obtain updated information: implicit, explicit, structural and historical. Implicit acquisition of information is based on observing the user working the system. Explicit information originates from direct dialogues (e.g. a questionnaire). Structural information is based on an analysis of domain-specific elements (e.g. if element A is prerequisite of an element B, mastering B implies knowing A). At last, the user's experience is examined to obtain historical information.

After acquiring the information, the data needs to be adapted to update the UM. In order to do that, the information is obtained by analysing information that originates from user responses, problem solutions and other user actions. These methods are analytical processes, defined as cognitive diagnosis (Self, 1993) – the process of inferring a person's cognitive state based on their methods and performance.

Analysis of user responses requires that questions have been well planned, structured into the related domain model topics. Also, incorrect answers need a careful study, as the domain knowledge can still be present (some fragments of the answer are correct). Examining the information originated from solving problems requires the system to know beforehand each possible rule (to solve the problem). Combining these rules with common errors and fallacies, the system identifies which steps the user is having trouble with. A full description for each method (including other actions) is available in Brusilovsky (1994).

3.3 User modelling techniques

Modelling involves inferring unobservable information about a user from discernible evidence (e.g. their actions). To perform this task, a system must deal with the uncertainty attendant to inferences about a user in the absence of complete information (Zukerman & Albrecht, 2001). Formation of a UM by observation of the user's actions involves a process of induction, where the system infers a model of whatever aspects of the user are of interest, such as preferences or objectives (Webb, 1998). User modelling consists in the process of creating and maintaining the UM. The forms that a UM may take are as varied as the purposes for which user models are formed. A UM intent is to describe (Webb et al., 2001):

- The cognitive processes that underlie the user's actions;
- The divergences between the user's skills and expert skills;
- The user's behavioural patterns;
- The user's characteristics.

There are two traditional methods to obtain a UM: either via a statistical analysis, or by implementing Machine Learning (ML) techniques. ML evaluations consist of splitting a data set into a training and a test set, using the first to learn the model and the latter to evaluate the model's performance (Zukerman & Albrecht, 2001). ML is taken to include automatic computing procedures based on logical or binary operations that learn a task from a series of examples (Michie et al., 1994). Statistical approaches are characterised by requiring an explicit underlying

probability model, which provides a probability of being in each class rather than a classification. Nonetheless, both fields have common objectives, with multiple scientific references merging or mixing them. In Breiman (2001), two arithmetical models (data and algorithmic) are identified, and Breiman defines the algorithmic model as “machine learning algorithms”. In Dhar (2013), ML and Statistical Models are identified as one, combining into the “data science” field. Many more definitions can be found, such as “Statistical learning” (James et al., 2013) or even with Neural Networks classification as a third field (Michie et al., 1994). This diversity happens due to the lack of an accepted standard definition by the scientific community. In this work, the most relevant techniques for user modelling are considered, and their classification into the statistical or ML field is not a concern.

3.3.1 Stereotypes

A stereotype is a collection of frequently occurring characteristics of users (Rich, 1979b).This technique is commonly found on AH (or user modelling only) systems, where new users are matched with a stereotype fitting to their characteristics – the small initial information is used to a large number of assumptions. As more information is learned about the user the assumptions are altered in accordance.

There are two types of stereotypes: fixed and default. In fixed stereotyping, users are assigned to a predefined profile according to their performance. Default stereotyping is dynamic, given that it adapts the initial characteristics (from fixed stereotyping) to more individualized ones as it observes and learns the user (Kay, 2000).

There are three elements in a stereotype: triggers, the information inferences and the retraction conditions. Triggers are what activate a stereotype, making possible to assign users to it. The information inferences are activated when a stereotype is assigned, and include the information that is supposed to be known. The retraction conditions establish when the stereotype is no longer valid. A practical example is presented by Froschl (2005): if a novice Linux user arrives in a learning system, the trigger “no prior knowledge about Linux” is activated, allocating the novice stereotype to this user. Then, the inferences are made, moving the known information to all novices to this user. When the user gets enough experience (over time), the “no prior knowledge about Linux” trigger is deactivated using the retraction facility, making the novice stereotype no longer valid for this user.

As described by Elain Rich (1979b), the stereotypes that the system contains knowledge are arranged in a directed acyclic graph (formed by the partial ordering relation “generalization of”). This is relevant since it removes the need to represent identical information in different stereotypes. Figure 5 shows an example piece of a directed acyclic graph containing the stereotypes for religious people. It represents the fact that there are many characteristics shared by Christians, regardless of denomination, and others shared by all religious people regardless of religion. This general directed acyclic graph can be used to focus the attention of the system on the events that are the most likely to be of significance.

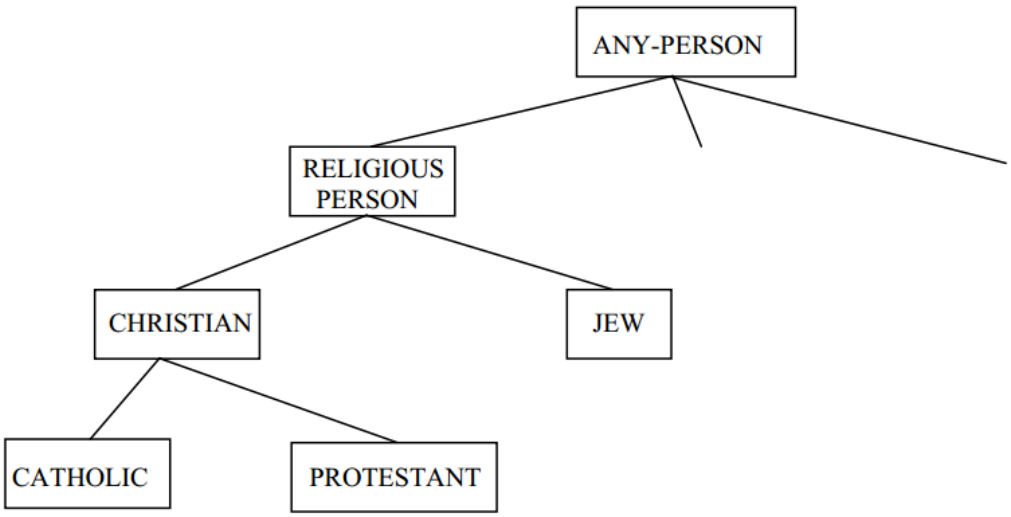

Figure 5 – Example of a graph containing stereotypes for religious people (Rich, 1979b)

The main difficulty with this technique is the amount of work needed to identify and construct appropriate stereotypes. According to Kobsa (1993), a modelling system has to plan tasks like user subgroup identification, identification of key characteristics and representation in hierarchically ordered stereotypes. To identify user subgroups, the UM developer must identify different groups within an expected user population. Triggers must also be identified in order to determine the relevancy of a learner to a specific subgroup, and the trigger number must be kept small due to performance issues (Froschl, 2005).

Although a stereotype may often provide highly appropriate and useful information about a system user it is, at the best case, an expression of a tendency and not the truth. This technique can be seen as useful to demonstrate user modelling and to provide a starting point on how to discover information about a system's population and how to exploit multiple UM (Rich, 1979b).

3.3.2 Bayesian Networks

Probabilities are a measure of uncertainty. The probability theory is a mathematical approximation that enables processing uncertain data. It proposes a value ***P(E)*** that consists in the possibility of the occurrence of the event E from a sequence of experiences with random events. If a certain experience is repeated a certain number of times, then we can be secure that the relative frequency of event E is nearly the same as ***P(E)***. There are multiple variations of probability distribution – as discrete or continuous probabilities – but, for the Bayesian theory, the most relevant is the conditional probability. It uses a value ***P(E|A)*** that represents the possibility of event E knowing that event A has occurred. This can only be used when we have the probability for event A. If there is more data available (e.g. event B and C), it should be included in the calculation ***P(E|A,B,C).*** Although the conditional probability ***P(E|A)*** is able to give the probability of an event E knowing that a prior event A happened, many times that value is known and the need is to discover the opposite, the probability of event A knowing that event E has happened. That is one of the advantages of the Bayesian theorem.

The Bayesian theory allows the continuous calculus of several possibilities as the events are observed.

$$P(A|B) = \frac{P(B|A)\,P(A)}{P(B)} \qquad (1)$$

Equation 1 relates the probabilities of A and B - ***P(A)*** and ***P(B)*** - and the conditional probability of A given B and B given A - ***P(A|B)*** and ***P(B|A)***, respectively. This theorem is the foundation of a Bayesian Network.

A Bayesian Network (BN) is a graphical model for probabilistic relationships among a set of variables that allows representing and reasoning about an uncertain domain (Korb & Nicholson, 2010). The nodes in a BN represent a set of random variables from the domain. A set of directed arcs connects pairs of nodes, establishing dependencies between the variables. The conditional probability distribution (CPD) is where the dependencies are represented. In this thesis, the term "independent node" is used when a node without incoming arcs is referenced and the terms "parent/child node" are used to mention the node where the arc starts/ends, respectively (Madureira et al., 2014b). As an independent node has no incoming arcs, the CPD is just the probability distribution of the variable. A child node - who has one or more parents – has a CPD that specifies a conditional probability for each value of the node given each combination of values from the parent nodes. The CPD relates all of the possible outcomes from the parent nodes to the probability distribution of the node (Baclawski, 2004). The constraint related to the arcs in a BN is that they must not create a direct cycle, assuring that the BN stays acyclic. To design a BN, multiple modelling decisions need to be made. The design of the network and the CPD for each child node is quite subjective, depending on the responsible for its creation and which process he decides to use. Usually, the first step is defining the variables of interest to the global problem. These are the variables that will be represented by the nodes in the BN. There are several types of nodes depending on the outcome. In this proposal, Boolean (yes or no) and ordered (e.g. values {beginner, intermediate, advanced}) nodes are used. After this step, the network structure is decided. The parent nodes are identified and the arcs connecting all nodes presented on the BN are defined. The main concern is the representation quality of the relations between each node: two nodes should be connected only if one affects the other, with the connection arc identifying which node is creating the effect (parent node).The final design step is to define the CPD table for each node. This procedure is done defining the probability that the child node will have for each possible combination from the values of the parent nodes. In this proposal, only the child nodes have unique CPDs, where the independent nodes are Boolean nodes with a yes/no outcome, equally distributed (0.5, 0.5). This final step can create a huge CPD table if a node has many parents (e.g. for *n* parent Boolean nodes the CPD table requires 2^{n+1} definitions).

There are several methods to implement a dynamic BN, ranging from structure variation to CPD changes, as well as both (known as full graph variation) (Baclawski, 2004). Bayesian Networks present multiple advantages over other techniques due to their flexibility, the capacity to deal with missing (unknown) values, the capacity to work as a framework for expert knowledge (in the CPD table) and due to its morphing nature, adapting itself to the user as he learns the system.

3.3.3 Neural Networks

An Artificial Neural Network (ANN) is an information processing paradigm inspired by biological Neural Networks (such as the human brain) and the way they deal with information. ANNs contain layers of simple computing nodes (as shown in Figure 6) based on neurons that operate as nonlinear summing devices. These nodes are related by weighed lines (connected nodes are

defined as neighbours) and learn by example – the weight is adjusted depending on the data provided during its training process (Dayhoff & DeLeo, 2001). The nodes have activations and their activation influences those of their neighbours.

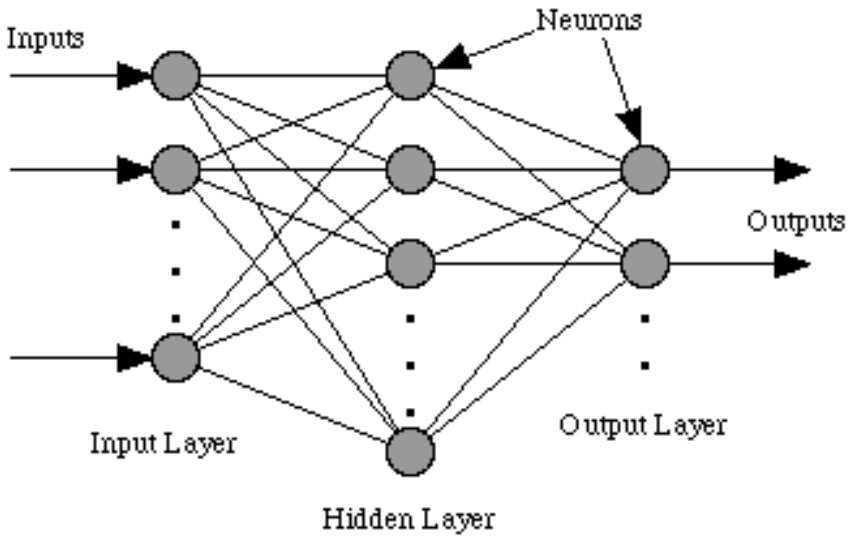

Figure 6 – Representation of a multilayer perceptron with an hidden layer (O'Connor et al., 2012)

The neurons work in parallel to solve a specific problem. From the user point of view, an ANN (after its training) is relatively simple: the network receives a set of input values and, in return, it gives the appropriate output.

There are several types of ANNs. The most popular include feedforward (signal only goes one way in the network) and competitive (neurons compete with each other to obtain the best solution). For each type of network, multiple activation functions can be selected (e.g. sigmoid, linear or step), increasing the complexity and possibilities of network structures. The type of ANN can influence how it operates, defining the relation between the way that the inputs are propagated in the network with the activation mechanism of the nodes (e.g. deterministically in feedforward networks or stochastically in Boltzmann machine (Anthony, 1997)). Different types of ANN topographies are suited for solving different types of problems (Krenker et al., 2011).

Current applications of ANNs involve real-world, complex problems, including the development of predictive models (to forecast future values of a particular response variable from a given set of independent variables), classification (such as decision making and pattern recognition) and data processing (e.g. clustering, filtering and compressing).

Regarding the user modelling research field, Neural Networks have been used for some time. As presented by Jennings (1993), the developed ANN has been constructed and modified as a result of articles read or rejected by the user. After being made, it analyses relevant words (that appear often) and feeds them to the network, strengthening the connection between nodes of similar frequency. With time, the most representative nodes dominate the network, creating a more accurate representation of users' interests. The proposed UM concept is a "long term" model (takes a long time to model users). However, it is a robust method that provides successful results. Another interesting study is presented in Van Heerden et al. (2008): ANNs

were applied to identify the students' success using input and output data from admissions into an undergraduate medical program. A total of 99 input variables from the UM were used to perform the calculations, and when all that data was accurately gathered, predictions were performed close to a 100% precision rate. Such thriving results had an impact on the institution selection process, particularly on identifying high-risk students.

ANNs are not superior to other techniques but, if used correctly, bring many advantages (Stergiou & Siganos, 2011). There are clear precedents on using ANNs to model any system's users; the challenge that still persists is the lack of a standard architecture (common to multiple modelling systems) that developers can take advantage of, which would decrease the required investment to implement these adaptive techniques.

3.3.4 Decision Trees

A Decision Tree is a structure that defines rules based on how to classify certain data into groups and allows the approximation of discrete-valued functions with disjunctive expressions (Frias-Martinez et al., 2006). Decision Tree learning is usually best suited to problems where instances are classified as value-attribute pairs (e.g. weather – rain), the target function has discrete output values (e.g. 1 or 2) or where the training data might contain errors or garbage (data without quality or relevance).

In UM, Decision Trees are mostly used to classify not only users but also any type of media in order to adapt the content – following a very traditional AH approach. Decision Trees are appropriate due to the vague nature of user data (Frias-Martinez et al., 2006). Another advantage over neural and Bayesian Networks is that this technique can have its rules designed by the author (as a BN) or using a learning algorithm (as ANN). However, it has some limitations such as high sensibility to small perturbations in data (a minor change can create a hugely unlike tree), a high dependency on the quantity of data (which is troublesome in UM context) and a high computational cost (in UM systems the adaptation should be smooth, instead of a burden to the overall system).

3.3.5 Other techniques

This section describes long-established methods to either obtain a UM or predict user actions. These techniques are fully investigated, being stable and correct to use but not providing several improvements found in their modern (and more complex) counterparts.

Linear models

Linear Models use weighed sums of elements to obtain a value for an unknown quantity. This technique uses a simple structure, being easy to learn, extend and generalize. Using an example where a Linear Model is chosen to predict user's rating for TV shows (similar to Raskutti et al. (1997)), the known values would be other users' ratings, and the weighs would be the similarity between those users and the user in question. The final Linear Model is the weighed sum of the ratings (Resnick et al., 1994).

Hidden Markov models

Similarly to Linear Models, Markov Models follow a simple structure. This technique employs the Markov assumption, which is related to the stochastic component of a system, as it needs to be memoryless - the next state of the system is independent of the past states given its current state (Natarajan et al., 2008). Markov Models have 2 variables: a state variable (similar to a Markov chain) and an observation variable related to the current system condition. The "hidden" name in these models comes from the initial assumption that the states are unobserved. Given a number of observed events, the next event is predicted from the probability distribution of previous observed events.

TFIDF-Based models

The TFIDF - Term Frequency Inverse Document Frequency – model is a weighing structure appropriate to find documents that match a user's inquire. It measures the similarity between two documents by inserting each one into a vector of weights. A TFIDF approach is, as expected, usually found on information retrieval systems (Salton & McGill, 1983).

Overlay Models

Overlay Models were commonly used on user modelling systems, particularly in the educational area. They were proposed by Carr and Goldstein (1977), and its function is to identify differences between the user's behaviour and an expert behaviour. It requires the creation of an expert's model that is modularized into specific topics. Each of these can be connected to a particular UM. The complexity of an overlay model is related to the structure of the domain knowledge, with the granularity having a major importance (Conlan et al., 2002). The UM is a set of hypotheses, each of which stating the confidence that the user has a certain skill. They are known as overlays to reflect that the models of the users are, essentially, a variation of an expert's model.

The advantage of using overlay methods is their effectiveness given that they are easy to implement. However, overlay models have some problems, such as the willingness of the user to follow another path (different from the expert), having other beliefs (not represented in the model) and not being able to predict user actions with limited knowledge.

Plan Recognition

Plan Recognition relies on analysing the user input and progress while using a system and compare it to a predefined sequence of actions (Li & Ji, 2005). If a likely match is found, the system then adapts its content in accordance to the predefined configuration.

Two types of techniques are used to distinguish the sequence of actions performed by the users: Plan Libraries and Plan Construction (Kobsa, 2001). Plan Libraries contain all predefined combinations of actions and the selection of the most appropriate plan to apply to a certain user is done based on the similarity of actions. This technique is particularly heavy on the computational work required, as it constantly compares each action to the ones available in the library. Plan Construction uses instead a backlog of possible actions and their respective effects and requirements. This allows the user plan to morph into a sequence with any action, and then the system tries to calculate the next action, adapting in accordance.

In general, Plan Recognition is not a widely used technique on large systems due to the fact that all possible (sequence of) actions have to be predicted beforehand and stored in a library that the system uses. Besides this, a specific user might have a specific sequence of actions that has not been properly foreseen by the system developers and, in this case, a wrong adaptation would occur. Given the other available techniques, Plan Recognition should be used only on systems operating on very small domains and where the user can only follow a narrow sequence of actions.

3.3.6 Current Trends

Ontologies

Although some AH systems present ontologies-based technology (even with some success, such as Zhang et al. (2007)), this type of technique is not yet considered as it should when developing new adaptive systems. Not only that, but current systems do not use any type of global standard, which would be solved using ontologies. However, if relating the UM and the adaptation from different systems using the same base ontology is an achievable challenge, the use of different ontologies is not. The problem of ontology mapping must be tackled if this method is going to be adopted as a standard for user modelling (Knutov et al., 2009).

Standards

One of the main concerns in every modelling technique is the chosen terminology to apply on the models and their description. While some current systems use XML or ontologies (Mohamad & Kouroupetroglou, 2014) to store their information, the majority relies on a proprietary structure that does not allow its reutilization. Using different terminologies affects not only the used storage type, but also the variety of information that a UM contains (e.g. its organization). This is justified by the diversity of areas where UM systems can be found.

Some efforts are being made to create a standardization capable of increasing the cooperation of researchers and developments teams that will expedite the evolution of this type of technology (Mohamad & Kouroupetroglou, 2014) but is still lacking major adoption. This refusal to use the proposed standards is related to the diversity of areas, to concerns with user privacy and the lack of rules for the interoperability of user models.

In the future, a strong standard (such as ISO) should be supported and properly constructed, regarding not only the reutilization of user models but specially the privacy issues. Ideally, legislation would be created to state the information exchange mechanisms to be used.

Virtual reality

One of the recent trends in technology is to capacitate systems (e.g. “simple” computer programs, games or a website) with virtual reality interfaces. This method allows the user to interact with the system content through a 3D representation that supports natural actions (such as looking around, picking objects or walking). This creates an immersive environment since it surrounds its users with multiple stimulus, providing a superior experience.

Even if this immersion approach can create more natural, appealing and even enjoyable systems, virtual reality still faces major challenges in order to obtain wide user acceptance

(Chittaro & Ration, 2000). One of the, if not the main, reason is the usability of the system: current system design guidelines do not deal with virtual reality. When the introduction of proper user modelling techniques to deal with this technology happens, it will increase the easiness and, therefore, the appeal and acceptance of this type of system to its users.

Privacy-enhanced user modelling

This topic is related not only to computational implications but particularly to legal and behavioural ones. Reconciling personalization with privacy has been an ongoing interest in user modelling research (Wang & Kobsa, 2013). Ever since user modelling systems started collecting user data to adapt its content that they have been subject to privacy laws and regulations if they want to respect its users' rights (Kobsa, 2007). Systems that are designed to work on an international level are particularly affected since different countries have different regulations. This type of system will have to follow different privacy laws. Besides, there is also the fact that users can choose to customize their experience via a configuration option, which includes their privacy level that can change from time to time (i.e. they can change their mind on that subject). This level of complexity makes it hard to deal with, advocating that current user modelling systems should be prepared to deal with this subject. Some advances are already being made (Wang & Kobsa, 2013) but there is still a long way to go. This growing concern can be related to the lack of a global standard, as previously described; developments in this field will always be connected to a standard on how to store user information and private data.

3.4 Summary

User modelling is an area that appears in 1979 which focus on adapting system's content to specific user needs in order to achieve a higher efficiency. Current systems are complex and have to deal with multiple users with distinct characteristics, so content adaptation is a key issue for modern system.

The UM is a structure that represents the system's beliefs about its users, providing the required information to adapt to their needs. The content of a UM is vital to its degree of success. An accurate UM must contain data regarding the user's preferences, goals, interests, domain knowledge and progress. Initially, the system can use several techniques (e.g. Stereotypes) to generate the first model. As time goes by there will be changes to user's preferences and characteristics, so a UM should be properly updated.

There are a number of techniques to create a UM. The well-known are Stereotypes, Decision Trees, Bayesian Networks and Neural Networks. Stereotypes consist in researching the most frequently occurring characteristics of users and try to match them to the current user (operating the system). A Decision Tree is a structure that defines rules on how to divide certain data into groups and are mostly used to classify both users and any type of media the system uses. Bayesian Networks present multiple advantages over other techniques due to their flexibility, the capacity to deal with missing (unknown) values, the capacity to work as a framework for expert knowledge (in the CPD table) and due to its morphing nature, adapting itself to the user as he learns the system. An ANN is an information processing paradigm inspired by biological Neural Networks (such as the human brain) and the way they deal with information where neurons work in parallel to solve a specific problem. Additional techniques, which are hardly found in current literature, include Linear Models, Overlay methods or Plan Recognition.

Current challenges in this area include the needed emergence of standardization (to increase the cooperation between systems with user modelling techniques), the use of Ontologies that can work across systems, the possible legislation on what type of information is appropriate to store about the user and, lastly, the recent Virtual Reality trend which requires new guidelines on how to design systems.

4 ADSyS Scheduling system

This chapter aims to describe the Adaptive Decision Support System for Interactive Scheduling with metacognition and user modelling Experience (ADSyS) system architecture. The developed modelling architecture described in this thesis is applied on ADSyS, so it is important to understand its structure. The ADSyS project aims to develop an intelligent system that is able to support scheduling tasks using machine learning techniques with distinctive learning patterns.

By observing the nature it is possible to conclude that many species are able to learn by observation or by experience. The ADSyS project aims not only to replicate this comportment, but also to incorporate social behaviours, as its core is based on a multi-agent system. ADSyS also has an intelligent and adaptable interface, further developed with the work presented in this thesis. The main goal of ADSyS is to combine human intellect with computational intelligence, creating an efficient method to solve scheduling problems.

This chapter is organized as follows: the first section presents a brief theoretical review on the most relevant themes to understand ADSyS; the second section explains the assembled architecture, illustrating the major modules that cooperate to solve scheduling problems; the third section reveals the results from sessions held in order to evaluate ADSyS with its current and potential users.

4.1 Theoretical review

4.1.1 Machine learning

Artificial Intelligence is a research field that has the objective of creating and developing computation methods that simulate human intelligence. One of the basic requisites to any intelligent behaviour is the ability to learn, as most researchers state that there cannot exist intelligence without a learning process (Bransford et al., 1999; Carbonell et al., 1983). Automatic

learning, known as machine learning, is one of the key areas of Artificial Intelligence, possessing one of the largest levels of development amongst its subdomains (Mitchell, 1997).

The construction of machines that are able to learn has stimulated humanity for a long time. This area looks precisely for answers on how to develop systems that automatically learn with examples and/or past experiences. As a whole, machine learning studies the fundamental laws that govern the learning process. For decades that theories and algorithms developed in this area have a huge impact on the comprehension of human behaviour aspects. Human cognition shows a massive potential to machine learning research given that a human being has the innate ability to, via observation and experimentation, acquire and organize new knowledge and even develop motor capacities (Carbonell et al., 1983). The study and computational modelling of the learning process in its multiple materialisations constitutes the main purpose of the Machine Learning research area.

Learning Types

Distinct learning systems may apply different knowledge strategies and representations. A cognitive system tries to understand the concepts of its ambient by creating a simplified model of that same environment. The building process of this model is named as inductive learning. A cognitive system is also able to organize its experience by constructing new standard structures. The creation of models and standards by a cognitive system is known as machine learning. The most common Machine Learning techniques can be classified into two groups, distinguished by their information type (Alonso et al., 2001): learning with examples and learning by observation. There is also reactive learning which is fundamentally different since it takes place almost spontaneously and only in response to recent events, so it should not be joint with the other two.

Learning with examples (Supervised Learning) is a technique where the cognitive system has to learn a function that, in reality, is a representation of a model. To the system itself a set of examples is given, including the proper output for each one of them. The system then needs to calculate the model based on the output values. Regarding evaluation, the model is composed using a subset of the original data (training set), and the remaining parts are used to evaluate the model.

Two learning processes are recognized by Supervised Learning techniques, namely Regression and Classification. Regression is associated with models that predict functions with continuous data; Classification is when the model predicts functions with discrete values. The most common Supervised Learning methods are: Instance based learning; Decision Trees; Bayesian Networks; Linear Regression; Neural Networks; support Vector Machines.

Learning by observation (known as unsupervised Learning) is a Machine Learning technique where a data set is served to a specific algorithm, creating a model (e.g. a Clustering algorithm). This type of learning is unique given that there are no output values when the data is provided to the algorithm. A set of input objects is reunited and treated as random variables, and the model is then built to a data set (Ayodele et al., 2009). The observation learning process can be divided into degrees of interaction with its exterior (Carbonell et al., 1983): passive observation or active experimentation. The model itself is generated after the algorithm processes the provided data set and describes the data characteristics. It is typified as a predictive model given that it is used to predict output values of a function to a specific input value. This model also provides quantitative information about the used data.

4.1.2 Dynamic Scheduling and self-organization

Scheduling function, in a manufacturing organization, is integrated in the global management decision level, considering the diverse organizational levels, their functional perspective or the integration of functions and business processes (Pinedo, 2008; Madureira et al., 2013). Most real manufacturing scheduling problems can be described as dynamic and extended versions of the classic Job-Shop scheduling combinatorial optimization problem. In practice, scheduling environment tends to be dynamic, i.e. new jobs can arrive at unpredictable intervals, machines can breakdown, jobs can be cancelled and due dates, release dates or priorities can change (Madureira et al., 2013).

Dynamic Scheduling (or rescheduling) could be stated as the ability to efficiently and effectively adapt, on a continuous basis, existing schedule plans according to external events or disturbances, keeping business performance levels. On real world manufacturing scenarios two categories of events could be considered (Madureira et al., 2013): events related with resources (machines breakdown, workers absenteeism, limit in production capacity and lack of correct tools to specific tasks) and events related with task (orders that arrive to the system too late or too soon, new orders not initially predicted, delivery dates changed or priorities changed).

In the last decades, there have been advances in research and application of Meta-Heuristics based approaches, with special concern on real-world problems solving. However, most studies have focused on developing several specific aspects of optimization and for static or deterministic scenarios. Several contributions have been proposed on literature to address different classes of manufacturing systems subject to different unexpected and imponderable events, such as machine failure; rush orders; job cancellation; due date modification (postpone or advance) delay in the arrival of raw materials or components; and changes in job priority. (Lee et al., 2003; Ouelhadj et al., 2003; Goren et al., 2012; Madureira et al., 2002, 2007a, 2007b, 2013; Varela et al., 2003; Madureira, 2010): However, scheduling is still having difficulties in real world situations and, hence, human intervention is required to maintain real-time adaptation and optimization. There is an increasing interest and exploration on decision support systems that utilize some of the ideas from autonomic oriented computing for handling problems in complex manufacturing systems and to identify mechanisms that make use of autonomous entities to solve computational problems through a self-organized behaviour.

Self-configuration requires support for reconfiguration from the business process design, which can be at the level of business process workflow or in the selection of individual atomic services. Self-configuration also requires adaptation and monitoring as an integral part of the business process. A reasoning engine is also required as part of the business process in a feedback loop.

Future Scheduling Systems have to display a significant extent of self-organization. The main goal of introducing self-organization, comprising self-optimization, self-configuration and self-healing (Madureira et al., 2007a) is to significantly decrease the operational costs by reducing human involvement in operational tasks while optimizing scheduling effectiveness and efficiency.

4.1.3 Multi-agent systems

The multi-agent paradigm is emerging for the development of solutions to complex and distributed computational problems. This paradigm is based either on the activity of intelligent

agents which perform dense functionalities or on the use of a large number of simple agents that can produce an overall intelligent behaviour, guiding to the resolution of difficult challenges.

Considering the complexity inherent to the manufacturing systems, Dynamic Scheduling is considered an excellent candidate for the application of agent-based technology. In many implementations of Multi-Agent Systems for scheduling in a manufacturing environment, the agents model the resources of the system and the tasks' scheduling is done in a distributed way by means of cooperation and coordination amongst them (Madureira et al., 2007a; Monostori et al., 2006). When responding to disturbances, the distributed nature of Multi-Agent Systems can also be a benefit to the rescheduling algorithm by involving only the agents directly affected, without disturbing the rest of the community which can continue with their work.

The multi-agent system structure developed in ADSyS is further explained in section 4.2.1.

4.1.4 Human-computer interaction

The forms of interaction between humans and computers have come a long way since their inception. However, the emergence and development of new technologies, accompanied by an increasing investment in research in this area, make the route of Human-Computer Interaction constantly evolve.

The progress of Human-Computer Interaction is verified not only in the quality of interaction, but also on the diverse interaction forms. The different research areas are looking to focus their attention on multimodality concept (rather than unimodal), intelligent interfaces (instead of interfaces based on commands) and active interfaces (rather than passive interfaces).

As a scientific area, Human-Computer Interaction is a multidisciplinary area that receives contributions from different areas, such as Psychology, Ergonomics or Artificial Intelligence, among others. Besides, this area focuses its research not only in the study of computer or humans but also gives special importance to the communication process between them.

The multidisciplinarity that characterizes this area of research is justified by the contribution from each of the involved research areas: the use of their knowledge for the identification and understanding of the human being limitations (Cognitive Psychology), the restrictions that existing technology imposes (Computer Science), the phenomena that the communication process comprises (Linguistics and Sociology), amongst others.

The importance of Human-Computer Interaction is related to the fact that even the most sophisticated computer system is useless if not properly used by its users. This argument relies on two main concepts that should be considered in the design of interactive systems: functionality and usability (Karray et al., 2008).

A computer system may be defined, in the context of Human-Computer Interaction, as what the system can do: the functions provided by the system should contribute to the achievement of the purpose for which the system was created. The functionality of a system is defined as the set of actions or services available to users. However, the value of a certain feature only becomes visible when the user is able to use it effectively. The usability of a system with a given feature is defined by the degree of efficiency and suitability in achieving certain goals for particular users (Karray et al., 2008).

Design (in the context of Human-Computer Interaction) is the attempt to create harmony between the user, the computer system and the services required to achieve a given performance, both in terms of quality and service optimization. Determining what is a good implementation of the interaction between the user and the system is very subjective, depending largely on the context.

The activities performed by the user present three levels: physical, cognitive and affective. The physical level determines the type of interaction mechanisms to be used. The cognitive level reads up on the user's method to understand and interact with the system. The affective level tries not only to make the interaction a pleasant experience, but also to encourage the user to continue to use the system. With this approach, a new attitude in the development of computer systems as emerged: the user should be the center of the development of computer systems.

4.1.5 Mixed initiative interaction

Personalization is a key aspect of effective Human-Computer Interaction (Shneiderman, 1998). Even if using a Mixed-Initiative (MI) approach does not primarily require a human (Hearst, 1999), it is one of the most used, with user acceptance to evidence its popularity (Al-Omar & Rigas, 2009). MI is defined as the mutual control by the system and the user in the communication between them. Its main goal is to deliver an ambitious system that is autonomous and able to recognize gains from modifying the interface and interacting with the user, doing it whenever it is beneficial. The other main advantage is allowing users to refine the interface according to their needs.

There are, effectively, two ways to adapt information without user input: either by drawing attention to certain content or by showing/hiding specific information. The variance is whether the information is available or hidden. The most known content adaptation techniques are as simple as sorting, zooming or scaling specific content (Knutov et al., 2009).

MI key principles and problems followed in this thesis have been defined by E. Horvitz (1999), with the key points being the significance of the value added by the automation, allowing efficient direct invocation/termination and considering the overall uncertainty about the user ambition.

4.2 ADSyS architecture

The ADSyS system is a scheduling system where communities of agents model a real-world manufacturing system subject to disturbance. Agents must be able to learn and manage their internal behaviour and the interaction with other agents, collaborating in order to obtain the proposed goal.

The prototype consists of four main modules (Madureira et al., 2014c): the integrated interface module, the scheduling module, the user modelling module and the dynamic adaptation module. Additionally, four interaction modules are considered (task editor, machines editor, order set editor and Gantt chart editor) which are responsible for the input data, i.e., for the definition of the scheduling problem and the visualization of scheduling results. The system’s global architecture is illustrated with the information flow in Figure 7.

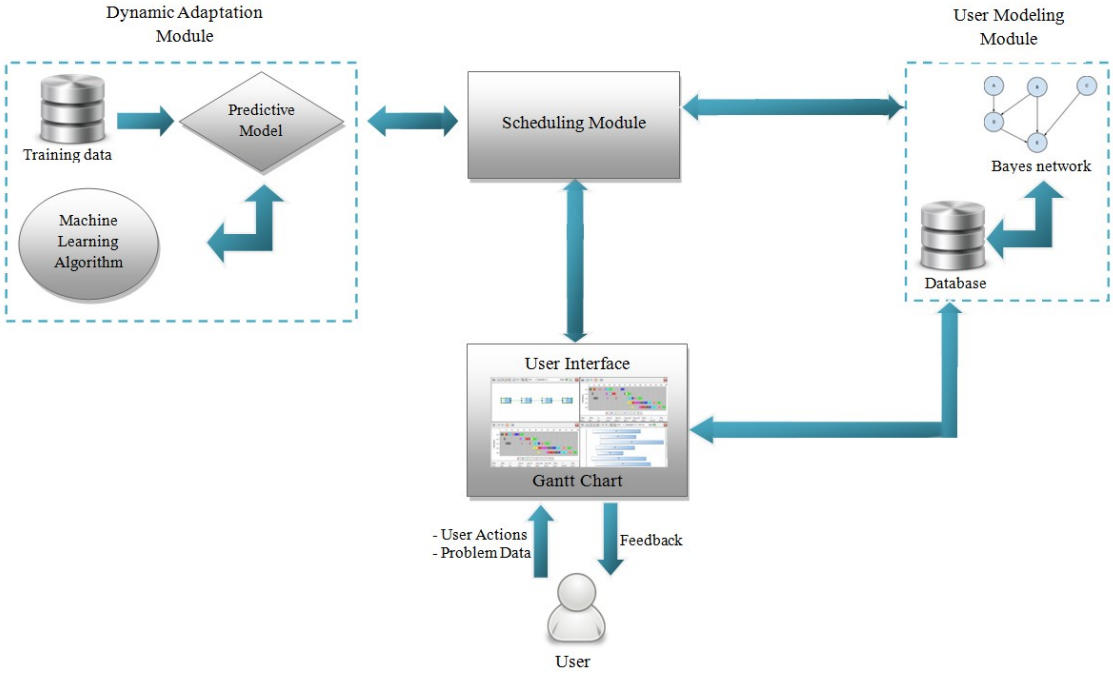

Figure 7 – ADSyS architecture (Madureira et al., 2014c)

4.2.1 Multi-agent structure

ADSyS, in its model, has agents representing tasks/jobs (Task agents) and agents representing machines/resources (resource agents) in a manufacturing environment. Additionally the proposed model considers a user interface coordinator (UI coordinator agent) agent responsible to coordinate single solutions obtained by each Resource Agent in order to obtain a global schedule for the original scheduling problem.

Figure 8 presents the proposed self-managed model, which has three distinct automatic agent types, designated as self-* agents (Madureira et al., 2007b): self-configuration agent, self-optimization agent and self-Healing agent. Considering the classification schemes proposed by Nwana (1996), the planned multi-agent architecture is hybrid since it combines two or more approaches in a single agent, that includes collaborative agents, collaborative learning agents and interface agents.

1) UI Coordinator Agent

The UI agent is responsible for a seamless communication with the user. This agent, apart from being responsible for the user interface, generates the necessary task agents dynamically according to the number of tasks that comprise the scheduling problem and also assigns each task to the respective agent. In the end, it collects the solutions from each Resource Agent and applies to it a repair mechanism in combination with a coordination mechanism (cooperation or negotiation). This agent is responsible for the dynamic selection of the appropriate Integration mechanism (IM) to use when a new task arrives. It is the controller of the proposed supervised Machine Learning module.

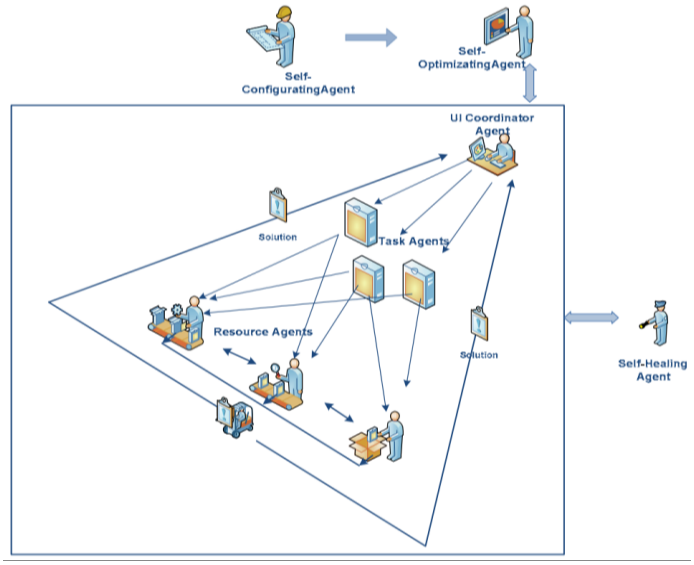

Figure 8 – Multi-agent system model (Madureira et al., 2014c)

2) Task Agents

Task Agents, which are collaborative agents, process the necessary information about each task, i.e. they are responsible for a pre-processing of the scheduling problem data. This represents the generation of the earliest and latest processing times and the allocation of operations to their respective Resource Agents.

3) Resource Agents

Resource Agents (also collaborative agents) are responsible for the scheduling of the operations that require processing in the machine they are supervising. They implement swarm intelligence methods (i.e. Particle Swarm Optimization and Ant Colony Optimization) in order to find the best possible operation schedules and, later, coordinate through cooperation/negotiation to improve the plans.

4) Self-Configuration Agent

The self-configuration agent is responsible for monitoring the system in order to detect modifications that affect the schedule, allowing the system to launch a dynamic adaptation. The agent works directly with the one described next.

5) Self-Optimizing Agent

This Self-Optimizing agent is responsible for the tuning of the Meta-heuristics' parameters, according to the problem. This agent receives the initial problem data (number of jobs/machines, its routing and operations attributes, etc.) or the change detected by self-configuration agent and automatically chooses the meta-heuristic to use, creating its self-parameterization. If some dynamism occurs, parameters may change in run-time. This tuning of parameters is made through learning and experience, since it uses a case-based reasoning (CBR) module. Each time a new problem (case) appears, the CBR uses past experience in order to specify the meta-heuristic and respective parameters for that case. When the new case is solved, it is stored for later use.

6) Self-Healing Agent

The Self-Healing Agent (autonomic agent) gives the system the capacity of detecting deviations from normal conditions and, proactively, takes actions to normalize them and avoid service disruptions. Since agents may crash for some reason, this agent provides one or more backup agents in order to ensure storage for the reactivation of lost or stuck scheduling agents with meaningful results, enabling the system to restart from a previous checkpoint (not a complete reset). With this agent, the system becomes stable and prepared to handle any problem that might occur.

4.2.2 User Interface

The user interface enables interaction between the user and the scheduling module in order to make possible operations such as the definition of meta-heuristic to be used (and its parameters), the results via Gantt charts or even the possibility to interact with it to modify results (e.g. incorporate dynamic events). These operations are done via specific editors (machine, task, order set and Gantt charts) which are presented ahead.

The proposed interaction model supports the definition of problems, parameter optimization techniques and visualization of the results. The system includes a multi-view feature, which allows users to view and interact with various objects simultaneously in different window layouts (1 to 4 windows, Figure 9). This feature offers advantages to the user due to not only being a technological innovation, but also because it presents a mechanism to compare and validate many of ADSyS system interaction objects at the same time. Since any object can be seen in the different layouts, this ensures that the system has a high degree of flexibility and information presented.

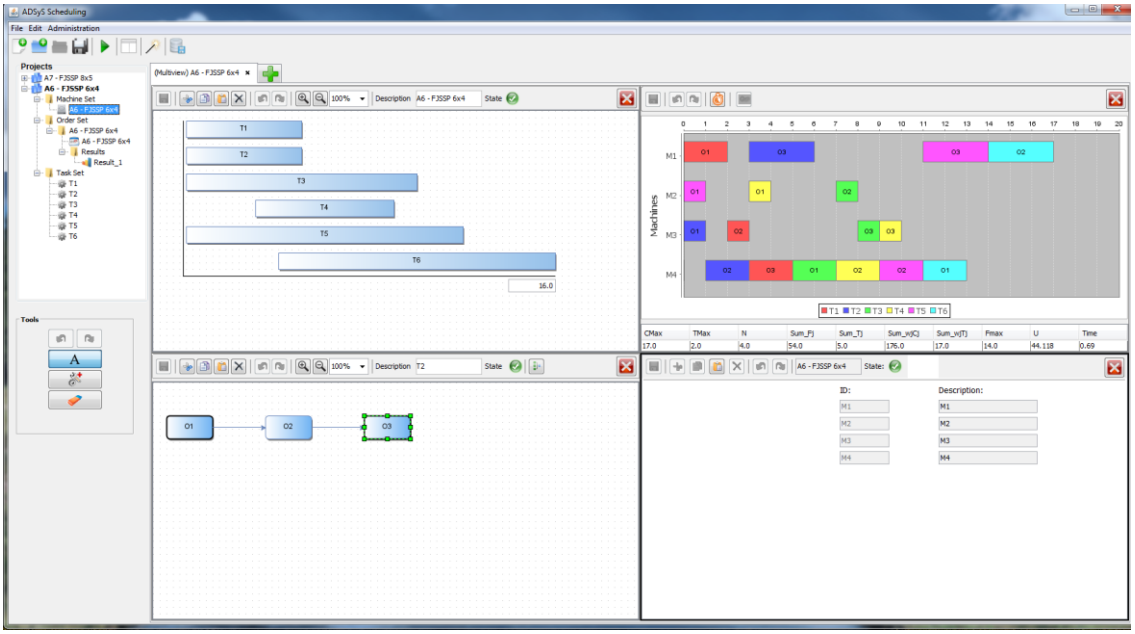

Figure 9 – ADSyS Prototype in multi-view

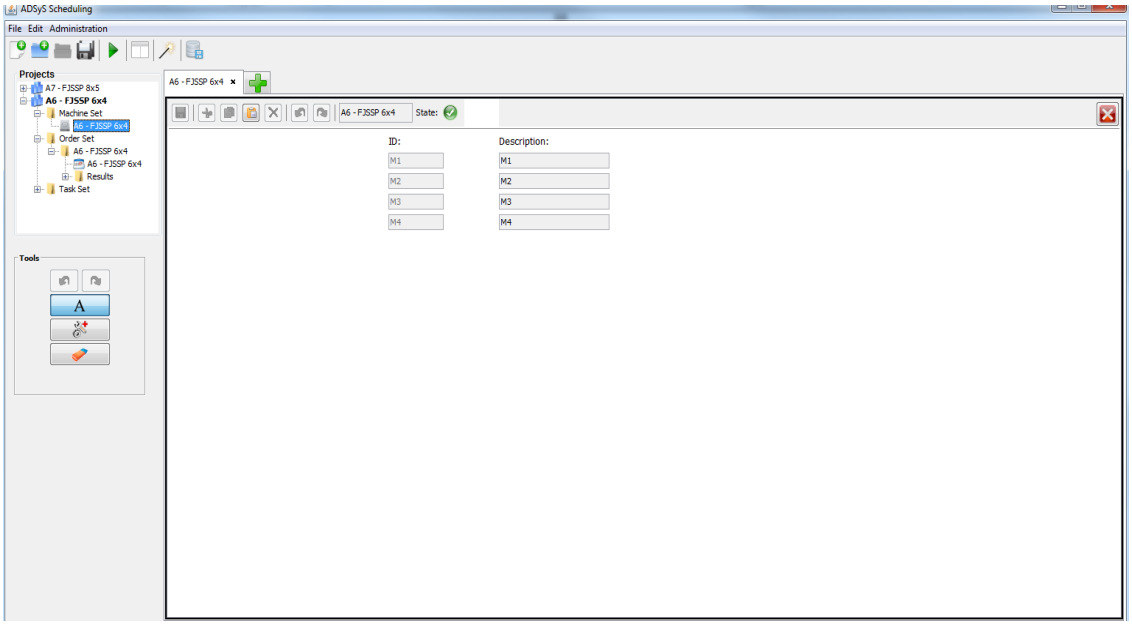

Figure 10 – Machine Editor

I. Machine Editor

The Machine Editor (Figure 10) purpose is to delineate the machines that are available during the operation definition. A machine is where the operations are processed. The number of machines is not restricted and they can be inserted and not utilized later, in the final plan. Machines are characterized by their identification and a description. The user is able to create, remove or modify any number of machines using this component. When the user finishes editing, the system provides feedback about some information that might be missing (e.g. missing description on one machine).

II. Task Editor

The Task Editor (Figure 11) allows the user to create a new task or edit an existing one. An operation is the basic unit of a task, since a task is defined as a sequence of operations. It is provided the possibility for the user to view and edit a task routing. A routing consists of operations and precedence. The user defines the sequence in which operations must be performed. An operation has the following characteristics (Piairo et al., 2013):

- Description: a descriptive name of the operation;
- Machine ID: identification of the machine where the operation is going to be performed;
- Processing time: processing time of operation on the machine.

The task editor allows the user to edit a sequence of operations. This sequence defines the order in which the operations should be executed. An operation may have another operation or a set of operations as precedent. Using this editor, the user is able to insert or remove operations and to define their sequence, amongst other options (e.g. copy previous operations and graph auto-alignment). To assure that the task was correctly defined by the user, a real time validation mechanism was developed. Each modification to the task is instantaneously

validated and, in case of failure, the wrong object changes its colour and a tooltip message is available, in order for the user to understand the cause of the error and how to correct it (Piairo et al., 2013).

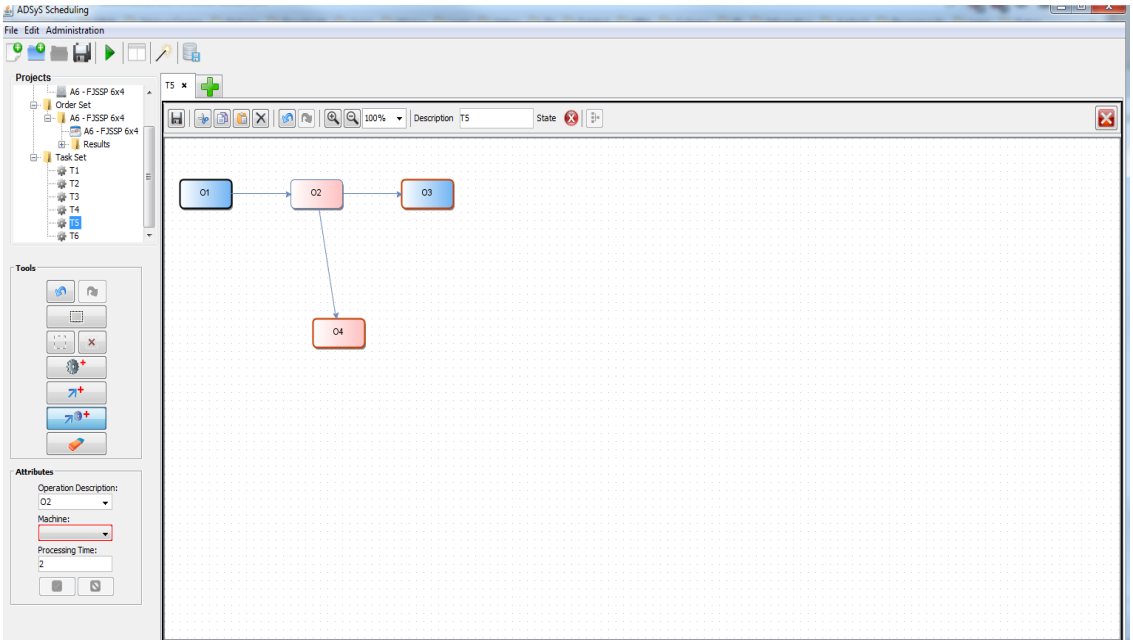

Figure 11 – Task editor

III. Order Set Editor

The Order Set Editor (Figure 12) permits to create a new order set or edit an existing one. A task is the basic unit for the definition of an order set. The Order Set Editor function is to define and edit the scheduling problem, taking advantage of a graphic interactive tool. A scheduling problem corresponds to a set of tasks to be executed during a given period of time. Each task has a set of characteristics: release date, due date, weight and quantity (Piairo et al., 2013).

The user is able to insert, remove, switch and modify the tasks. There are two types of movements that can be applied to the tasks: horizontal and vertical. When moving horizontally, the task release and due dates are modified, maintaining the duration. If the user wants to resize the task, one of the dates (either due date or release date) is changed, along with the duration. In vertical movements the user is capable of switching the position of one or many tasks simultaneously. Other operations that switch the positions of the tasks are removing a task or inserting a new one.

Similar to the previous module, the order set editor also has real time validation to guide the user through the definition of the scheduling problem. Several verifications are made (e.g. maximum due date) and contextual feedback is provided to the user.

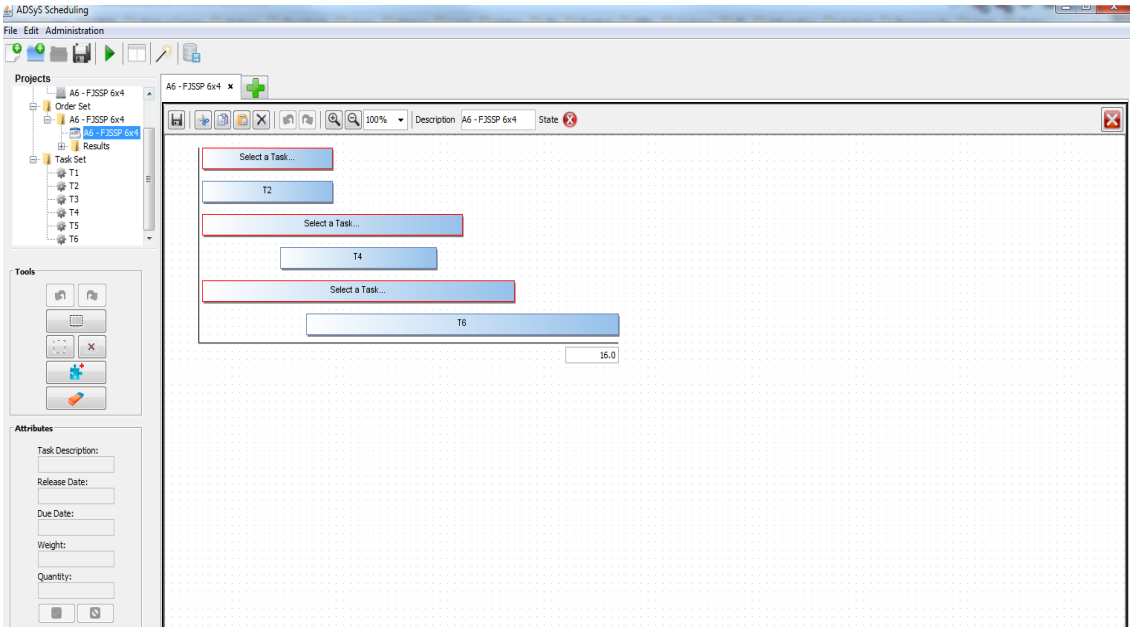

Figure 12 – Order Set Editor

IV. Gantt Chart Editor

The results of the scheduling module are provided to the user via a Gantt chart (Figure 13). The results are included in a tool that allows the user to see and edit the operations in the scheduling plan. The interaction gives users the option to tune the plan reflecting their needs. The user can alter the plan by performing horizontal operation movements. The lateral transfers imply a modification of the position of an operation in the given machine, allowing the delaying or anticipation of that operation. When a movement is performed, the system does two validations: operation overlapping – two operations are not able to be executed at the same time by one machine – and task constraints – an operation is only executed after its task release date and once the corresponding, if any, preceding operation has been processed.

4.2.3 Scheduling module

Scheduling Module is the scheduler itself; constructs a scheduling solution to the problem instances throughout a MH technique. It is able to deal and incorporate system dynamisms (new rush orders, job cancellation, jobs attributes modification, etc.). The scheduling problem defined in (Madureira et al., 2007b, 2002; Madureira & Pereira, 2010) is decomposed into a series of Single Machine Scheduling Problems. Each Single Machine Scheduling Problems is solved by a metaheuristic, a local solution is obtained and later, through cooperation, in order to overcome technological constraints, a global schedule is achieved.

4.2.4 Dynamic adaptation module

This module employs Supervised Learning techniques to predict the best IM when incorporating a new task that arrives in a Dynamic Scheduling problem. The key idea of this Machine Learning

approach is to create a classification model, where the input features are mapped from the input variables into the output variable via a supervised classification algorithm.

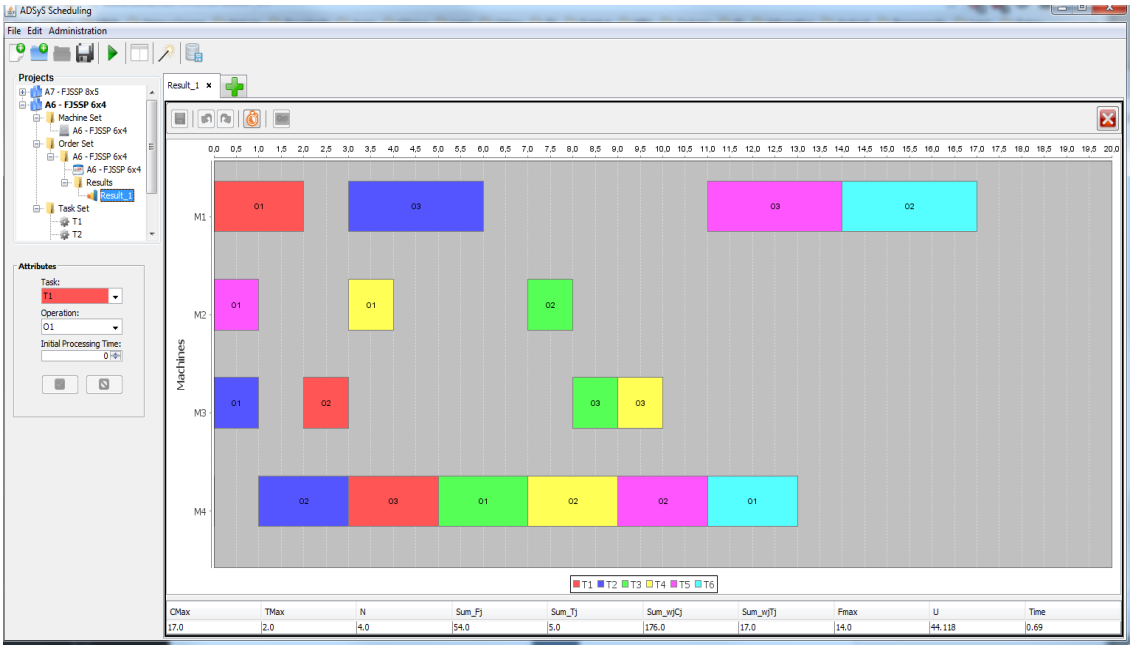

Figure 13 – Gantt chart editor

This approach is composed by three major components: a database, a Machine Learning algorithm and a predictive model. Any intelligent system needs a large database to be trained and tested. The database contains previous collected training data which will be used as an input to Machine Learning training, and that produces the Machine Learning classifier. This classifier has the responsibility to predict the best IM to use in a given problem. A study was needed to determine which classification technique was more appropriate to use. Based on that study (Madureira et al., 2014d), the decision was made to select the Decision Tree algorithm as the classification technique.

The working flow of this approach is relatively straightforward. When a new task arrives in ADSyS, the information related to the current scheduling problem and the new task is sent to the Decision Tree based classifier, which will then return the predicted IM (such as earliest due date or greatest priority first, amongst others). After that, the system can use the predicted IM to incorporate the new order in the current scheduling plan. However, each classification is subjective, and the user might not agree with the prediction of the system. So, the user must have the final decision about what happens in the system. It asks the user if he wants to use the predicted IM or select another, and then the system will act accordingly to the user desire. This approach increases the usability of the system (Madureira et al., 2014b).

In a classification system the successful classification rate depends on several factors. One of the most important factors is the quality of the training data used to create the classifier. Therefore, in order to increase the accuracy of the classifier, the system verifies if the case already exists in the database (training data). If not, the new case is added and a new predictive model is created. As result, the training data is increased with more real-world examples of

rescheduling problems. Hence, the learning process of the classification algorithm can continuously self-improve and, therefore, the classification will be more effective and efficient.

4.2.5 User modelling module

The user modelling module was introduced in the ADSyS prototype with the purpose of easing the learning curve of new users while boosting the productivity of expert users. It consists in multiple components cooperating in order to characterize the user behaviour and analyse it, using the gathered information to deliver a customized system experience. The Database (Figure 7 on page 36) is where the data from the interaction with the prototype is saved. The BN (Figure 7) is where the mathematical analysis is applied to the stored facts, with the outcome being the level of expertise of the user. The main differences between each level are the quantity of helps, the automation and the interface. At a lower level the system will offer multiple helps and suggestions, ranging from tutorials to full explanations of the functionalities. A user without expertise will also have more support from the system (e.g. filling complex forms with appropriate values). At the upper level, an expert user does not need the same amount of helps and is able to decide the level of automation provided by the system. The prototype also offers a configuration option to each user, assuring that each one has the optimal experience with the system (e.g. an expert user who wants multiple helps with a lower level of detail) with a higher usability.

The work related to this module is the reason for the research and development presented in this thesis. With the exception of the dynamic adaptation module, which was developed in parallel, all of the former modules where implemented in ADSyS before the development of the user modelling module. The specific components and their workflow are described in the respective subchapters (below chapter 5) and include not only the module itself but also the ADSyS adaptation required to perform the computational study (e.g. creating a result export option).

4.3 Summary

ADSyS is an intelligent system that is able to support tasks scheduling. The development of this system was done based on several research fields, from Machine Learning and Dynamic Scheduling to Multi-Agent Systems and Human-Computer Interaction.

The ADSyS architecture is composed by multiple components, which include the scheduling module (responsible for allocating the tasks) and the user interface module (which contains multiple editors for scheduling related tasks), amongst others.

This thesis proposal of an architecture for user modelling was built as part of ADSyS, so it is imperative to know the base system in order to understand the user modelling process.

5 Architecture for User Modelling on Scalable systems

In order to adapt any system to its users, a proper architecture should be planned. In this chapter, the developed modelling architecture is presented. It is composed by a user information unit, a mathematical structure able to classify users and the technique to use when adapting content. The user information unit is responsible for capturing every part of relevant interaction between the system and its users, and also contains the database (and its structure definition) which stores that information, delivered afterwards to the classifier. The mathematical structure is the user classifier; it is responsible for analysing users and allocating them into one of three roles: beginner, intermediate or expert. With the user role defined, a proper technique to adapt system's content is required. In this work, a Mixed-Initiative approach is detailed. This technique is based on allowing both the user and the system to gain control in the communication between them (e.g. the system can infer advantages and interact with the user and the user can directly ask for suggestions).

Regarding the user classifier two structures are selected: an Artificial Neural Network and a Bayesian Network. As previously stated, one of the objectives of this thesis is to analyse and compare both structures. A detailed guide on how to plan, prepare and train them to deal with user information is provided.

5.1 Motivation

Being able to adapt to its users is crucial to any modern system (Kay & McCalla, 2012). However, designing systems that are able to truly understand its users are hard, as subjects are constantly changing and become more complex to read.

In today's world, global competitiveness requires that organizations adopt agile techniques (even for their scheduling). Reducing the time required by a user to obtain the desired result, either via suggestions or by providing the required tools earlier, which can be a difference maker; that is the motivation for user modelling: to increase system's usability. A higher

usability directly translates into higher effectiveness rates, requires less time to accomplish specific tasks (more efficient to use), is easier to learn and understand and creates a satisfactory feeling on its users, making them more adept of reusing the system. A complete review on user modelling purposes and techniques is available in chapter 3.

Despite the huge developments in user modelling, one of the main challenges remains: finding a common approach for integrating user profiles that support different users within individual implementations. With this proposal, a descriptive guide on how modelling techniques are implemented into ADSyS (chapter 4) is presented. This serves not only as a proof of concept on a complex system but also as a starting point for future adapting systems.

5.2 User Information

5.2.1 Personas

In order to create the user information unit there was a need to collect and organize information on potential ADSyS users. With that in mind, a personas technique was selected. User Personas are referred as an approach to user modelling that improves the usability and user experience in a system. It is one of the most used user-centered design techniques. A Persona should be a precise description of a user and what he wishes to accomplish (Hix & Hartson, 1993).

Following Pruit (2006; 2003), the first steps were performed with the objective of collecting data about potential system users to create the User Personas. Three primary Personas and a non-primary persona were created and are further explained in the next sections. Each primary Personas represents a class of users that share the same needs. The non-primary persona was conceived due to the dynamic environment of user experience growth from the UM module.

Primary Personas

Table 1 – Adam Persona

Adam Tuff	Description	Experience
	21 years old, is a student in the IT area. Had a class about scheduling and is trying to learn more about the subject	Minimal, only knows some small concepts that help him understanding the goal of a plan
	Scenario	**Requirements**
	Will use ADSyS in order to finish a university assignment, where he needs to find a possible solution to a given problem	Will use ADSyS in order to finish a university assignment, where he needs to find a possible solution to a given problem

The Adam Tuff Persona (Table 1) represents the class of users that have less or no experience with the system. To this class, the system needs to adapt and become more welcoming. A user that has never worked with the system will have a hard time discovering every function and knowing how to operate with them, hence the need of some sort of a welcoming guide. This type of tutorial should explain the correct use of every tool and clarify, when not used correctly,

what was done wrong. There should also be sufficient explanations throughout the system, via dialogs or tooltips, which help the user adapting to ADSyS.

Table 2 – Clara Persona

Clara Terlford	Description	Experience
	41 years old, works in the industry area. Has never used a specific system to design scheduling problems	Is used to work with scheduling plans - knows the rules and terminology, but has no knowledge about this system
	Scenario	**Requirements**
	Due to some changes at her workplace, Clara will now start working with ADSyS when there is a scheduling problem or decision to be made.	Wants concise helps and suggestions. Needs an explanation for system specific details (e.g. order/task set editor, MH parameterizations). Desires an easy access to the tools she wants

The Clara Terlford Persona (Table 2) embodies the user category of someone who is used to work with scheduling plans during the professional career. She has the knowledge of the constraints and terminology that a scheduling problem have but she has small to no experience (at all) in using ADSyS. This user class needs concise, specific helps in order to be more proficient with the system. The other main need is to know the system-specific details and definitions (information that may be provided via the specific helps). This necessity is due to the fact that MH definition and parameterization, although known in general, has a different terminology in multiple systems, particularly the abbreviations. With the presented features Clara is able to make a swift transition from her background in scheduling to being skilled using ADSyS.

Table 3 – Leonard Persona

Leonard Hart	Description	Experience
	32 years old, is a scheduling expert. Is used to deal with the specifics of a scheduling problem and in using ADSyS to do so	"Maximum" experience, already knows the system and its terminology
	Scenario	**Requirements**
	Leonard has been an avid user of ADSyS for multiple purposes, whether it is to obtain a batch of results for a specific problem or to introduce a real-world change to a previous solution (using the Dynamic Mode)	Wants to use the system in an efficient way – needs advanced features (e.g. Wizard, Multiview, Dynamic mode). Does not want all helps to interfere with his activity – configurable helps and automation

The Leonard Hart Persona (Table 3) personifies an expert user, not only in scheduling but also in using ADSyS. This user category needs a different approach from the previous one, since their desire is not having more helps or a guiding approach, instead preferring a quick access to advanced system tools that allows them to be more efficient using the system. The Leonard Persona also wants the system to be configurable, letting it take the decision on global definitions (e.g. if helps should be automatic or if it should appear warning messages). System messages targeted at Leonard should also be concise, offering the option of further detail only at the request of the user (Hix & Hartson, 1993).

Secondary persona

The three primary personas all have a static profile, with constant values that do not represent a real learning curve that typical users will have while using the system. This fourth Persona is someone viewed as an "on-off" type of user, who goes through the three previously described BN classes: beginner, intermediate and advanced. It is not a primary Persona because there is no need to develop a new interface just for her (Cooper, 1999; Blomkvist, 2002), as she will be balancing between the three main Personas.

Table 4 – Sarah, the secondary persona

	Description	Experience	Requirements
Sarah Bray	37 years old, has some scheduling knowledge but does not have to use it regurlaly	Has no specific experience with ADSyS or scheduling systems, but is a prolific PC user and has worked in scheduling before	A dynamic and flexible system that can adapt to her evolution, growing with her and easing the learning curve
	A	B	
Scenarios	Started using ADSyS on a daily basis, due to some changes in her routine. Has gained knowledge and experience, being now classified has a scheduling expert	Has not worked on a scheduling problem for a long time. Although retaining semantic knowledge of the system over time, she tends to lose syntactic knowledge. Must be treated as a beginner	

The Sarah Bray persona (Table 4) is therefore a secondary Persona, who has a similar amount of knowledge as Clara, but does not have a clear scenario in which to use ADSyS. That means that she can use the system rarely, and be closer to a beginner classification, or she might happen to start using the system continuously at her workplace, making her a lot closer to an advanced, Leonard-like profile. Sarah was created to reinforce the feeling that although each primary Persona is static, a real user of the system is not. Sarah requirements are that the system must handle the evolution of a user (either gaining or losing knowledge of the system) in the best possible way.

The personas approach presented in this subchapter is the base for the profiles carved in the user classifier: ADSyS is able to categorize every user into one of three profiles based on the developed Personas. This is detailed in the next subchapter.

5.2.2 User classification

In ADSyS, system users are categorized into one of three profiles. These were developed based on the personas study described earlier. They go from beginner (level 1) to intermediate (level 2) and advanced (also known as expert; level 3). Each level has different necessities; if the system is built around those needs, each group of users in a level will be more satisfied than if the design is done only targeting one level, or all of them simultaneously (Brusilovsky, 2000). A schema representing the default types of help provided to user groups is presented in Table 5. The types of help are suggestions (e.g. a possible improvement on a scheduling plan), information (e.g. explain how a specific tool works), warnings (e.g. when a move makes the overall plan worse) or errors (e.g. illegal moves). By default, on level 1, all helps are provided, while level 3 users only see error messages (Table 5).

Table 5 – Default helps by User Level (Madureira et al., 2014b)

Helps/Level	**1**	**2**	**3**
Suggestions	X		
Information	X	X	
Warnings	X	X	
Errors	X	X	X

The beginner role aims to represent someone who has no experience with the system and scheduling systems. A user of this type will require the maximum amount of helps from the system, ranging from tutorials and better explanations for the functions to effective suggestions, amongst others. There is a tendency to reinforce the feeling that the user is an amateur when providing these helps and that should be avoided, instead treating the user as an intelligent one.

The intermediate user considers people who have some level of familiarity with the system (and scheduling systems) or to intermittent users, which use the system sporadically. If the user only has experience with one of them (e.g. knowledge about scheduling plans but not regarding the system) the BN will infer differences and the system will adapt to the user needs. The intermediate class of users prefers an easy access only to the tools they actually need to use and not to the more advanced features, only knowing that they are there in the case they are required at some future point.

An advanced profile represents someone who is an expert in scheduling concepts and is fully familiarized with the system. This represents someone who not only has developed an instinctive feel for the interface and who looks for very specific information in helps, but also wants to define some level of intelligent automation, stipulating where it should act and each specific parameter of the action.

The main differences between each level are the amount of helps, automation and the working interface. For a level 1 user more automation is required, as the user is less experienced. Some examples can be filling formularies with default values and presenting a tutorial on how to use the system. The opposite is level 3, where many of the helps and confirmations are disabled. To reach the last level, the user needs to be an expert, so there is no reason to present detailed explanations and to require a confirmation before each modification. Also, level 3 profiles should have more advanced options present in the interface as compared to the beginner, who could not have the knowledge to use them, so they should be hidden in a menu. This guarantees that all users have the most appropriate functions in the interface, and the ones not so relevant stay hidden, but accessible, under a menu. The more the system is adapted to the user profile, especially interface and the interaction, the more successful the classification will be. Nonetheless, each classification is subjective, leaving the user classifier with the task of inferring the probabilities for each user type. The system must have a way to personalize the overall configuration (e.g. a level 3 user who wants to confirm all modification or a beginner user who wants no tutorials and desires shortcuts to advanced functions), preventing a possible classification mistake and guaranteeing that each user has a custom experience with a high level of usability, exactly mirroring their needs. Once again, it is important to reiterate that all of the predicted behaviours are based on the executed investigation regarding the personas that would use ADSyS.

5.2.3 Database and information capture

To properly classify system users, a suitable database (DB) structure and capture technique are required. The DB contains the data about users' interaction with the system. In order for this repository of information to be accurate, a capture method also needs to be implemented, maintaining each fragment of relevant data every time the users perform a significant action. This capture consists in two key steps: saving all of the operations that users perform and retaining their level of expertise when performing (Madureira et al., 2014b). The first is attained by implementing the observer pattern in each event and dialog that the system has. The second is accomplished by analysing the parameterizations that the user selects and by comparing each plan before and after the user modifications.

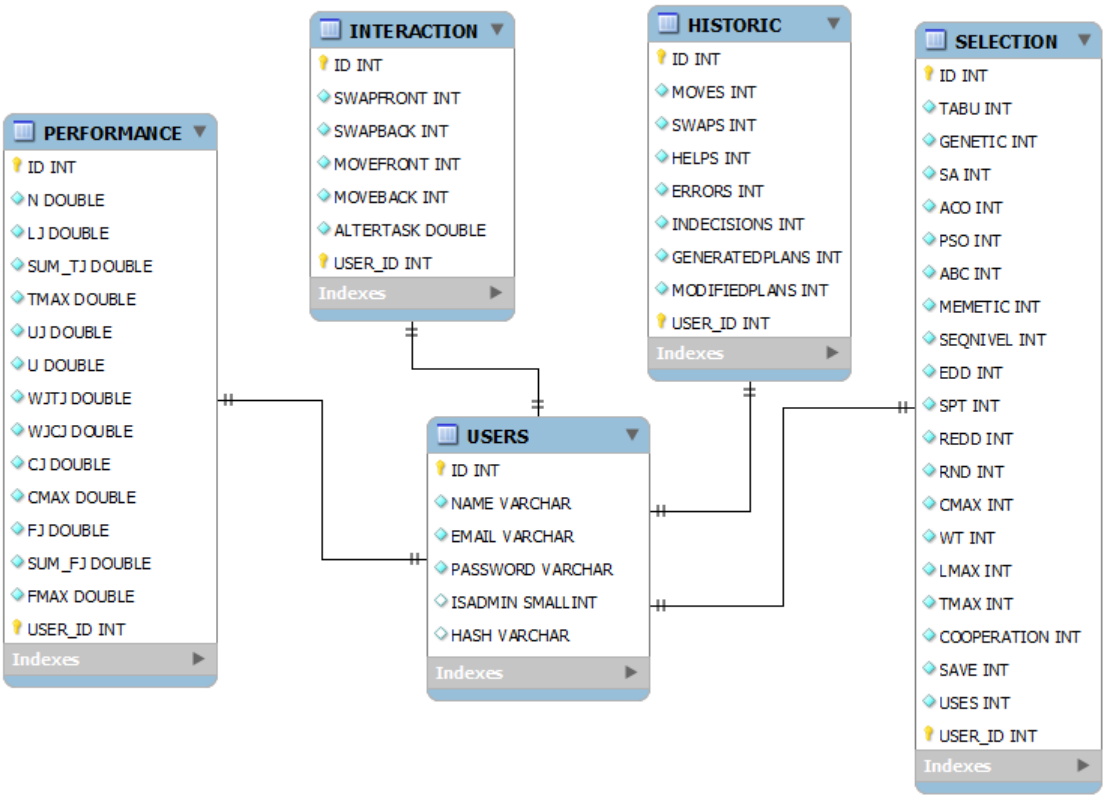

Figure 14 – Database for user modelling

The built DB (Figure 14) consists in six tables (Madureira et al., 2014b), each with a distinct use:

- Users – holds the information about the accounts in order to differentiate them;
- Performance – retains every possible measure used during the before and after plan comparison;
- Interaction – contains data that characterizes the user interaction with a modified plan;
- Historic – holds values for the basic actions that can be done (e.g. using the "undo" option will count as one indecision action and Historic.GeneratedPlans is the field that counts the created plans);

- Selection – is where the parameterization options, used during the generation of a new plan, are stored;
- Helps – keeps an historic of each help type that the user needs or does not use.

Except in the Users table, each field serves as a counter for several actions: almost every field relates to other tables columns to infer conclusions (e.g. the number of operations that were anticipated – *Interaction.MoveBack* – Is analysed against the total number of moves - *Historic.Moves* – to determine the tendency to move an operation back). The relevant information about each possible interaction with the system is kept in the DB, to be used later by the classifier and the MI approach. The DB is, by default, in a state of ignorance: each field starting at the lowest value possible (usually 1.0). This is a safety measure, preventing mathematical problems (e.g. divisions by zero) with the user classifier.

The more the system – specially the interface and the interaction – is adapted to the user profile, the more successful the classification will be (Brusilovsky, 2000). Nonetheless, each classification is subjective (as the user classifier could fail for a variety of reasons): ADSyS offers users the chance to personalize the overall configuration (e.g. an expert who desires to see all possible suggestions or a level 1 user who does not want any tutorial), preventing eventual classification mistakes and assuring that each user has a custom experience with a high level of usability.

5.3 User classifier

A classifier is a system that performs a mapping from input data to a category. In ADSyS, the constructed user classifier is a mathematical structure that is responsible for scrutinizing users and allocating them into one of three roles with an increasing level of expertise: beginner, intermediate or expert.

The selected structures to classify users are a BN and an ANN. Both are well established techniques and are commonly used to perform numerous functions (e.g. medical diagnosis (Amato et al., 2013; Nikovski, 2000)). The advantages for selecting this methods instead of others (e.g. Decision Trees) are described in the literature revision, on subchapter 3.3. It is important to note that both classifiers can be used due to the fact that all relevant variables are independent; if not, a Bayesian classifier would not be possible to apply in this context.

This study uses two structures with the purpose of later investigating which one could be more appropriate, presenting a robust real-world comparison (in chapter 6).

5.3.1 Bayesian Network

There are several methods to implement a dynamic BN, ranging from structure variation to CPD changes, as well as both (full graph variation) (Baclawski, 2004). In this proposal, the dynamic BN used is one where its probabilistic structure varies in time. Sensibly, the probabilities in the CPD table are subject to modifications over time, but the BN graph structure is static.

The proposed BN has been designed to classify the users in 3 different levels, as described before. The graph structure is represented in Figure 15 (developed with the Microsoft MSBNx toolkit (Kadie et al., 2001)). The BN has a total of twenty nodes, with 13 being independent.

UserType, the outcome node – the last descendant and the one that classifies the user – has three states: Beginner, Intermediate and Advanced. During the early development of the network, all nodes were connected to the outcome node, creating a huge CPD table. In order to increase the perceptibility of the graph, logical subgroups were introduced. This consists in creating nodes that have the independent nodes as parents and that are, themselves, parents of another node. This goes on until the outcome node is reached. The proposed limit for a node number of parents is four, although the majority of the subgroup nodes have only three parents. Also, in order for this rule to be effective, each node (except the last) included in the proposed BN only has one outgoing edge – although a node can have many parents, every parent only has one child node – and the connections should be established following some business logic (e.g. the proposed BN combines the analysis of a single task *j* completion time, C_j, and the overall completion time, C_{max}).

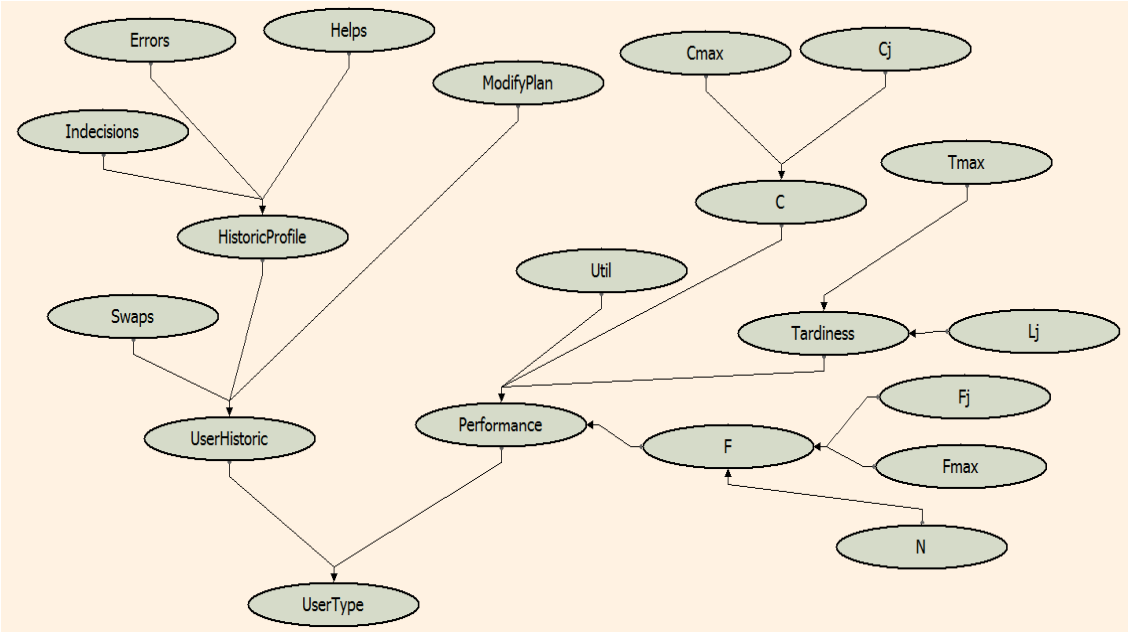

Figure 15 – Bayesian Network graph structure (Madureira et al., 2014b)

The 13 independent nodes are the ones which have their CPD revised to reflect the experience of the user. They start with an even probability distribution – 0.5 for the 2 Boolean outcomes – and these default values are combined to create a result establishing the user as beginner. Nonetheless, it is only after a certain quantity of experiences that the BN is used. The non-independent nodes – each node that has a parent – have a unique and static CPD, defined beforehand in accordance to the expertise outcome expected from the BN. The definition of the static CPD values is a gruelling task, as it requires a meticulous analysis of the user information (chapter 5.2) to create, on a trial and error method, proper tables that are able to classify the user correctly. A static CPD table from the developed BN is presented in Figure 16: the Performance node receives the values from the Util, C, F and Tardiness nodes and based on the static definition (the Yes or No red and yellow columns, respectively) it creates a value, ranging from 0.0 to 1, on the topic of the user's proficiency to improve a scheduling plan.

Parent Node(s)				Performance		
Util	C	Tardiness	F	Yes	No	bar charts
Yes	Yes	Yes	Yes	1,0	0,0	
			No	0,8	0,2	
		No	Yes	0,8	0,2	
			No	0,6	0,4	
	No	Yes	Yes	0,8	0,2	
			No	0,6	0,4	
		No	Yes	0,4	0,6	
			No	0,25	0,75	
No	Yes	Yes	Yes	0,75	0,25	
			No	0,4	0,6	
		No	Yes	0,4	0,6	
			No	0,3	0,7	
	No	Yes	Yes	0,4	0,6	
			No	0,3	0,7	
		No	Yes	0,2	0,8	
			No	0,0	1,0	

Figure 16 – Performance node (non-independent) static CPD definition

To infer the new values for the independent nodes, the BN uses the database stored information – each time that the DB is updated, the nodes CPD reflects that change. Using the C_j, let us assume that the database has data containing 76 modified plans and 43 C_j improvements for a specific user. In that case, the conditional probability that the user improves the C_j, knowing that he will modify the plan, is 0.605 – the network is learning since the starting 0.5, giving a more accurate reflection of the user. This happens for each independent node, with the variant being the formula used to calculate each probability (e.g. the node Errors uses the captured information from the interaction with the scheduling diagram, not the number of modified plans like C_j).

5.3.2 Artificial Neural Network

As expected, the ANN has been developed to output the user classification, varying from beginner to expert. The ANN has 3 layers, including the input and output ones. The input layer has a total of 13 nodes, one for each stored variable in the DB related to the user's interactions.

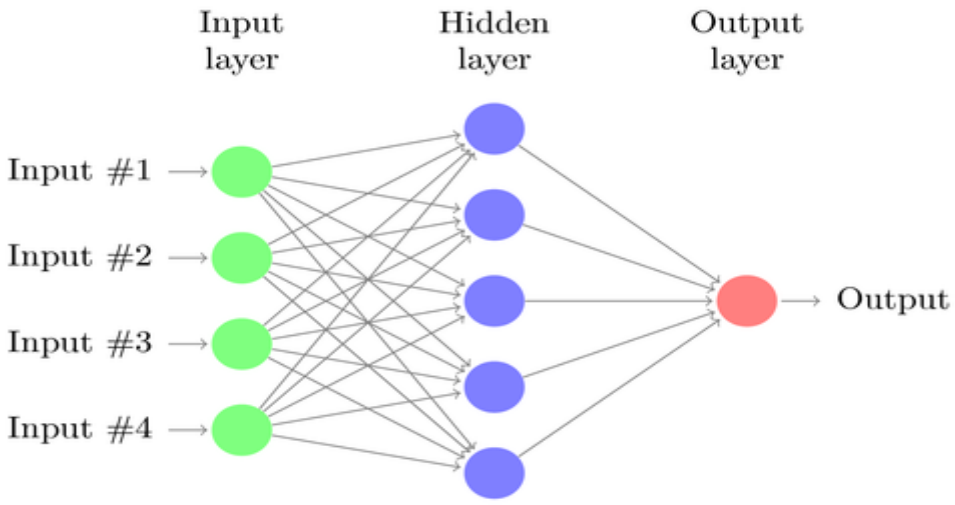

Figure 17 – Sample of a similar, low-scaled ANN to the one developed

The hidden (middle) layer, and its number of neurons, is related to the complexity of the problem (the more complex the problem is, the more neurons are needed). It is responsible for helping the rest of the Neural Network learn. The middle layer is composed by 26 nodes. This number was achieved taking the number of variables that affects user classification and doubling it. Nonetheless, it is not such a straightforward process as just multiplying; it requires a long study and test where several values - layers and neurons - are attempted until a proper precision rate is achieved (the training phase).

In order to accurately categorize system users, the Neural Network has to have appropriate weighs in each node. This is obtained by training, with supervision, the ANN: using a data set which contains input data and its expected output. The network performs multiple runs and varies node's weighs until it is able to classify properly the training data.

One of the issues of ANN training is the size requirement of the training data set: if there are not enough cases and/or they are not sufficiently different, the network will not reach its goal. However, using a big dataset creates an initial large amount of work, but is pays off in a proper network definition. In order to assure a proper training, combinations from every field stored in the DB and their respective classification were created: for each DB variable, an appropriate value was defined and combinations from possible values for the other variables were created (imitating real user cases). This process is executed for each DB variable and each appropriate value that every variable needs. For example, a DB variable such as C_{max} was fixed with values of 0.0, 1.0 and important real users values, such as 0.42, 0.71 or any appropriate value verified on real-world cases. Then, and for each of the fixed values, all the other variables was also set with their fixed values and the correct output was defined. In the end, the combinations were inserted in the data set, which was sent to the ANN. The node weights were locked when the network accuracy was bigger than 98% (correct predictions rate).

The developed network follows a feedforward structure: the information moves only in one direction (from the input nodes to the output node, going through the hidden layer) and connections between the units do not form cycles. In order to train the network, the resilient propagation learning heuristic, created by Martin Riedmiller and Heinrich Braun (1992), was used, allowing an efficient and transparent training process. The selected activation function was a sigmoid function, which is appropriate given that the network does not have to deal with negative values.

To infer the user classification, the ANN utilizes the information stored in the DB. Pragmatically, each time the system requires the user level (usually after login), the ANN obtains the most recent user information from the DB and transmits it to the input nodes, receiving the user level of expertise as output. To prevent classification errors, a MI threshold value is defined, stating the minimum amount of information needed (e.g. number of sessions, generated or modified scheduling plans) before starting to use the classification.

In spite of appearing as a waste of time, for both the user and the system, to recalculate the user expertise every time it is needed, the ANN implementation used in ADSyS follows a singleton pattern (only one active user for each running instance) which guarantees a fast calculation, reducing to a minimum the waiting time required between reclassifications (Cunha et al., 2015).

5.3.3 Divergences between methods

Even if both the ANN and BN provide a similar output (the user's level of expertise) they have considerable differences when being developed - not considering the theoretical contrasts already addressed. The biggest disparity happens right from the start: on a BN the developer has to define not only the network structure but also the CPD table for each combination of parent nodes. An ANN for instance requires significantly less work. To create a proper ANN for user classification it is only needed to state the number of neurons and layers (including hidden ones) that the network will have, and even that process can be automated (Fahlman & Lebiere, 1990)).The weights for each neuron are then defined in the learning process of the network, when the example cases are given to it. This creates the equilibrium and justifies the fact that neither method has a great advantage over the other: when looking at the amount of work to reach similar results, ANN have advantage if there is already data to be properly train. A Bayesian approach is better if there is no previous work, as creating a significant case base to train an ANN is very demanding.

Aside from the amount of work needed, the decisive factor to select the appropriate method is the quality of results, i.e. which network is a better classifier. In order to be able to properly investigate this matter a thorough study has been planned. The study consists in taking a set of real user cases (knowing their level of expertise) and making the networks classify each one of them. In this specific case, the network that is more precise, i.e. that knows the users better, is the best one to use. This comparison is part of the results that can be found on chapter 6.

5.4 Mixed-initiative

The purpose of knowing users and classifying each one into one role – beginner to expert – Is to adapt ADSyS to their needs: shape ADSyS navigation structure, presentation and content to the knowledge level of its users, their goals and navigation model. The system priority is to provide more tools to advanced users and observe and support starting users. There are two major types of helps: anticipate probable errors and provide useful recommendations. The first one is mostly present when modifying a scheduling plan. The system reads independently the actions and their combination to present the user with an appropriate suggestion, according to the context. In spite of using a good amount of non-obstructing messages to interact with the user, the help is not restricted to this form of communication. The second type of help is presented whenever the user is creating or modifying a plan. If creating, the system is able to infer, using the DB and the classifier, the appropriate parameterization for the new plan and automatically fills the forms, proposing to the user the adequate values. When altering the generated scheduling plan there the system makes automatic changes, using system resources, such as the DB or the user classifier, to infer the appropriate adjustments to do (e.g. if a task is scheduled by the user to start before its arrival date the system automatically prevents it and explains to the user the issue).

Every type of help provided to the user is given according to the DB information and, consequently, the UM and the user classification. To an expert in scheduling plans (advanced user) the automatic changes are mostly removed and messages are kept to a minimum, as being categorized as an expert means that the user is able to recognize most openings for improvements. To a real beginner who does not know the system, every available help is given, such as presenting explanations for errors to warn about poor parameterization selections and

expanding tooltips with supplementary information about how to use each tool, providing a friendly environment where the user is able to learn and, step by step, gain knowledge about how to use the system, creating a mental model on how to operate it. A typical roadmap for a new user would be to absorb information from helps or explanations that the system offers, evolving into an intermediate user that does not need the same amount of guidance and requires less automated task. At last, the new user grows into an experienced one, being able to improve plans without the need for helps and rarely making any misstep. In spite of the existence of a typical roadmap for each user, all of them are able to customize the type of experience they are looking for (e.g. an expert using explanations to understand new ADSyS tools or a beginner not desiring further improvements to a generated plan suggested by the system). This guarantees that the end users have exactly the experience they need and desire, further increasing the usability of ADSyS.

5.5 Summary

To be able to adapt a system's content to its users a proper user modelling architecture is required. This thesis proposes an architecture that is composed by three main components: the user information unit, the user classifier and the mixed-initiative.

The user information unit is subdivided in the user category and the database that stores the information. The categories range from beginner to expert, based on a previous Personas study (Madureira et al., 2014a). The database stores every possible evidence that can be used to allocate the users into their respective category.

The user classification, into one of the three categories, is performed by a mathematical classifier. Two classifiers were developed: a BN and an ANN. The BN is composed by 20 nodes and each one contains a handmade CPD table. The ANN relies on 3 layers, each with a custom number of neurons. While the ANN structure is mostly independent from the developer, in the BN it is required to define not only the structure but also the CPD table distribution for each combination of nodes. A Bayesian structure should be advantageous when starting from scratch, with no previous work. If there is already a case base of examples that can be used for training then an ANN would be the fastest, and possibly best, approach. However, there will always be exception to this theoretical rule; it should be always analysed case by case.

The user classification (provided by the classifier) is, at last, used to adapt the system content to the user needs and preferences. This customized experience is provided to the user through a MI approach, ensuring that both the system and the user have control in the communication. That means that the user is not only assisted when the system considers appropriate but is also able to ask the system for support. This method assures a high usability on the proposed user modelling architecture.

6 Discussion of Results

Results are the ultimate objective of a scientific research. This chapter consists of the required observations and measurements to fully understand the relevance of the decisions and techniques described in the previous chapters. To identify which user classifier (ANN or BN) is most appropriate for user modelling (at least on ADSyS) a statistical study has been accomplished, comparing the accuracy and efficiency of both networks. In order to discover if the proposed classification and respective system adaptation is beneficial for system users two evaluation sessions were held, each with relevant profiles of ADSyS users (from students to scheduling experts). The statistical study compares both classifiers and their proficiency to conduct the ADSyS user classification. The evaluation sessions results analyse the impact and benefits of the adaptation performed by the system to the user expertise. An analysis of the overall results and the full architecture is finally presented.

6.1 Classifiers: Bayesian Network vs Artificial Neuron Network

In order to evaluate if there is a categorisation disparity between using a BN or an ANN to perform the user classification a study was performed. This study consisted in obtaining user cases and pass them via each network. Afterwards, the percentage for all user profiles, from beginner to expert, was saved. At last, the final user classification from the BN was compared to its respective ANN counterpart, creating the final results shown in Figure 18 which presents the percentage of users on each level for both networks.

The case base used to compare the networks was obtained both from real user cases and via random generation. The random generation consisted in creating arbitrary values for each field used to classify the user. This ensures a substantial case base but introduces some artificial values which would not be found in a real-world scenario. However, this fact is not considered as negative due to the possibility of exploring the capacity of both classifiers to adapt to those extreme and fictional values.

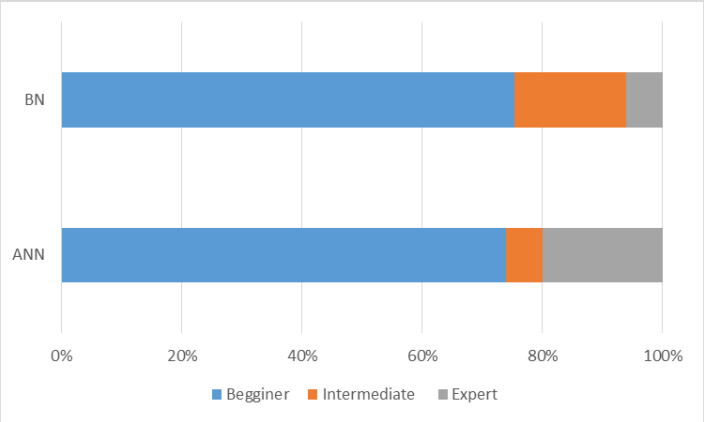

Figure 18 – Percentage of users on each classification by the Neural and Bayesian Networks

As seen, in Figure 18, the networks present mostly similar classifications. Regarding the number of profiles classified as beginner, the networks present a difference of 1%. Such a minor difference concludes that both networks are well prepared to discover and deal with users that are new to the system, which is both the main focus of the presented architecture and the reality of ADSyS users. On the subject of users assigned to the intermediate and expert levels, the networks present a 13% and 12% disparity, respectively (with the 1% discrepancy being attributed to that same difference on the beginner level). This variation shows that the studied networks have a contrast of classification when users gain expertise, even if only on a small percentage. After a proper analysis of each case with a classification disparity the variation can be attributed to the generated instances: while the ANN was trained and kept adapting itself even to such extreme cases, the static BN is more conservative, prepared to deal properly with real users but not handling well the generated instances – close to 10% of the total cases, near the disparity percentage between networks.

The result from this study is in accordance with what was expected from the literature and empiric perspective. If properly design and trained, both networks provide accurate classifications and can be used interchangeably. From a computation time position, both networks are very fast (on ADSyS), presenting their classification in a non-noticeable timeframe, so the required time to calculate the user expertise is not a relevant factor.

Pragmatically, when there is a desire to introduce user classification into any system, a BN should be used if there is no previous work done as it only requires the definition of the CPD table, feasible after identifying the user information (as done on chapter 5.2). If a case base of information on how users interact with the system (and the associated proficiency) already exists, it might be better to implement an ANN, as it can be enhanced to recognize any type of user without the need to manually define the node weighs.

Specific case analysis

On this subchapter a specific test case is analysed in order to present how the networks classify each user instance. Table 6 presents the values for each field required to classify the user, presented on chapter 5.2.3. In summary, from *swaps* (tendency to swap tasks in a plan) to *mod* (tendency to modify a plan) is analysed the user knowledge on using the ADSyS system. The other values, such as *Util* (machine utilization rate) or C_j (tendency to improve the makespan of

a certain task j) are used to evaluate the user expertise on scheduling plans. The values range from 0 to 1, with 0 indicating that it is not verified (e.g cannot improve a plan, does not commit errors) and 1 denoting that it is sure that the user will do it.

Table 6 – A test instance used to compare the networks

Field	Swaps	Ind.	Errors	Helps	Mod.	Util	C_{max}	C_j	T_{max}	L_j	F_j	F_{max}	N
Value	0.50	0.33	0.47	0.61	0.12	0.31	0.06	0.09	0.23	0.59	0.17	0.49	0.13

From an empiric standpoint, the correct classification for this user would be a beginner role, as he does not have a reasonable improvement rate on key fields, such as C_{max} or Util, and still has considerable values regarding errors and need for helps.

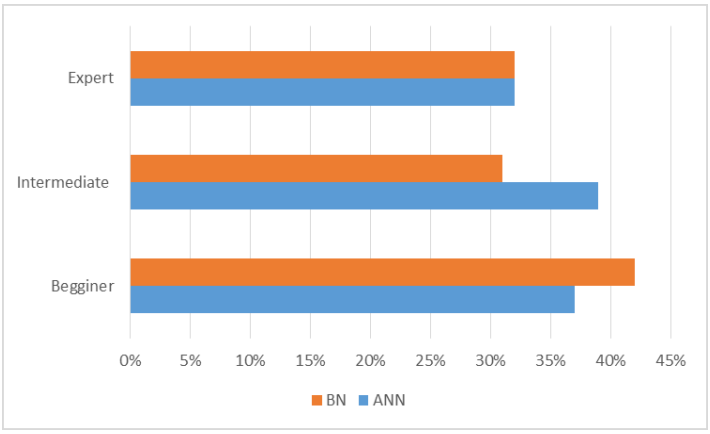

Figure 19 – Classification of the case on Table 6 by the Neural and Bayesian Networks

The classifications from the ANN and BN for this case are presented in Figure 19. The BN esteems a probability of 32% for the user to be an expert, 31% to be on the intermediate level and 42% to be a beginner. The ANN considers probabilities of 32%, 39% and 37% for the user to be on an expert, intermediate and beginner level, respectively.

Even if the BN provides the more accurate classification, the ANN is very close, only separated by a 2% deviation. It is correct to predict that the ANN would classify the user correctly as it observes and learns how he uses the system. While observing, the inputs to the ANN (and the BN) would be slightly modified each time the classification is required. This mechanism allows not only the possibility of the user knowledge evolving (e.g. intermediate to expert) but also to easily circumvent possible classification inaccuracies.

6.2 Usability evaluation

In order to analyse the state of the developed features two usability evaluation sessions were held. Both sessions were prepared and conducted with users having the same conditions.

The sessions included the following parts: An introduction, an initial survey, the prototype testing and the final survey. The introduction had the intention of contextualizing the

participants in order to understand the purpose of the evaluation session, the tasks to be performed and the type of questions presented, so that the responses were reliable and valid. The initial survey was designed to identify the profile of the participants and evaluate their knowledge about scheduling systems, and was filled out before the testing phase. The next part was the actual test, where participants had to perform a sequence of tasks in order to explore the system. This test is a great tool to observe and collect data about the interaction between each participant and the system as they follow a real-world scenario of using ADSyS. At last, the final survey aimed at obtaining the participants opinions about the previous test.

During the sessions each participant had its own script and a working ADSyS workstation, which is available in the Appendix of this thesis, in Portuguese (since it was the language it was conducted in). At the beginning of each session a brief presentation about the purpose of the evaluation was made, as well as an explanation of the tasks to be performed and how the surveys should be filled. Next, it was requested to fill out the initial survey. After this, the ADSyS system was presented, in order to demonstrate how and when all its features can be used. It consisted in a brief demonstration on how to perform basic operations and an overview of the modules, so that the participants knew the basic navigation procedure through the system. Also, it was given to the participants the opportunity to make questions about the prototype or other factors about the evaluation.

Due to the complexity of ADSyS it was given to each participant a scheduling problem already prepared to work during the test. The test consisted on a set of fifteen tasks to be performed by each participant. It asked them to change several predefined values in the given scheduling problem and to create and add new tasks and operations. As they advanced, the participants were asked to generate a new schedule plan and compare it with the previous one in order to visualize the changes that had occurred. For each participant the time needed to complete the test was recorded and, as expected, the participants who took less time were those who had experience with scheduling problems using ADSyS.

After all participants finished testing they were asked to fill out the final survey. This survey asked the users about several interaction aspects, about the ease of use and valuableness of the features offered by the system.

The first evaluation was held with a group of 9 users, representing 90% of usability problems (Virzi, 1992). They either had some scheduling knowledge (both on scheduling systems and theory) and/or had already worked with ADSyS (only one user had no previous knowledge). The aim of this session was to test ADSyS with its potential users, discovering their opinions on the system interaction, including provided assistance via UM and mechanical tools, and ideas of improvement. The gathered users for the second session were master's-degree students who were undertaking classes related to scheduling theory and systems. This type of users contrasts with the other session and allows to study the impact of the assistance provided to new users. The results of this session were schemed, beforehand, to focus on the UM module and how the system adapts the content to users (when they start and has they grow), as the audience expertise level and experience would be low. The first evaluation is clearly more important for the global ADSyS evaluation than the second one. The last session was held mainly to fix the lack of completely inexperienced users in the first one.

The most relevant information, resulting from the first evaluation session and the adaptation results from the second sessions, are presented and analysed next. The responses acquired from the surveys were classified in a Likert scale which is a measure of values between two

opposite attributes; with the maximum value being 3, the medium value 0 (neutral) and -3 as the lowest value.

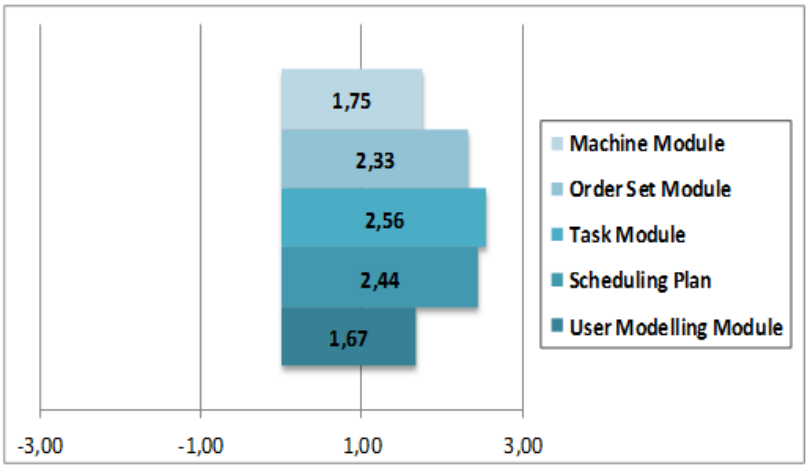

Figure 20 – Overall rate for each ADSyS Module

Figure 20 depicts the overall rate given by the participants for each ADSyS Module. The task editor, order set editor and the scheduling plan editor got an average rating over 2 units, representing an overall very positive assessment. Each classification proves that the ADSyS interface and interaction is quite good, evidencing the good design of the system and the interface adaptation, via user modelling.

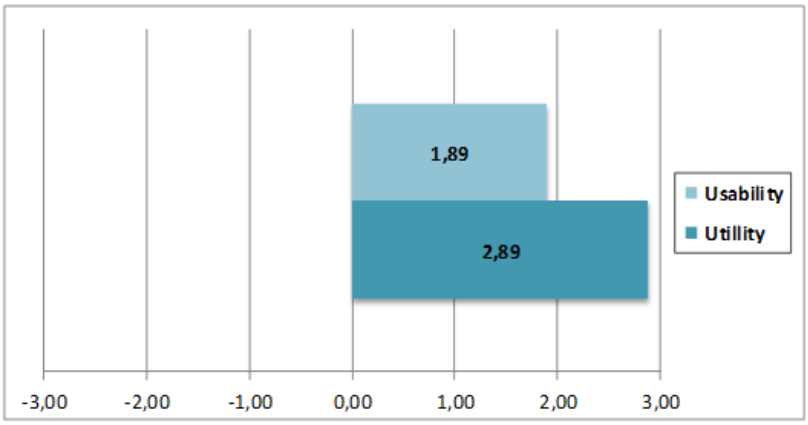

Figure 21 – Classification of dynamic mode feature

Figure 21 shows the overall rating of the dynamic mode feature in terms of usability and utility. Average scores show that participants are of the opinion that the presence of the dynamic mode feature in the ADSyS prototype is very useful. The good evaluation concerning the usability of this module is a good factor to demonstrate the usefulness of the user modelling module, responsible for adapting the interface and tools to the user. This is especially relevant here due to the complexity of the dynamic mode, diminished by the system adaptation to the user's expertise.

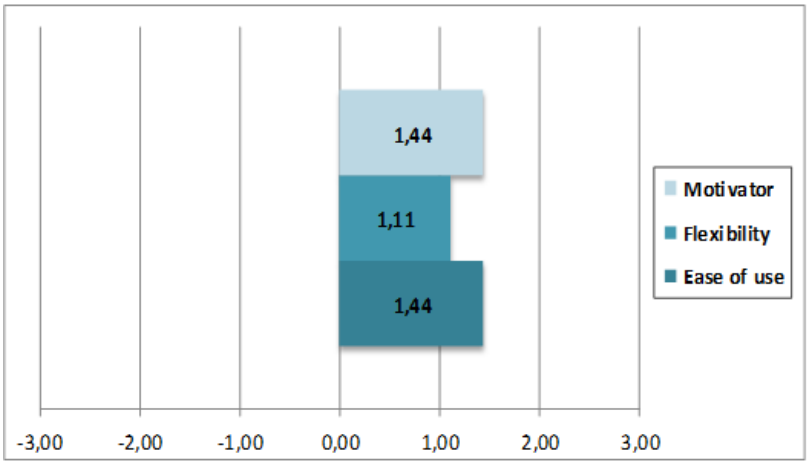

Figure 22 – ADSyS usage opinion

Figure 22 depicts the participants' perceptions about ADSyS, according to motivation, flexibility and ease of use. Participants found that the system was fairly motivator and easy to use. Once again, this information confirms the value added by the user modelling adaptation, as it composes an easy system to use, even for new ADSyS users.

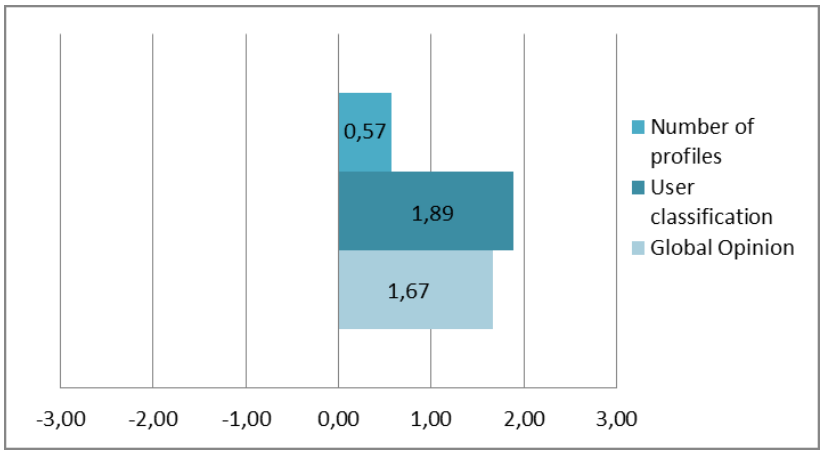

Figure 23 – ADSyS Personas evaluation

In the questions related to Personas classification, illustrated in Figure 23, it was asked if the number of profiles and the user classification was appropriate in accordance to their expectations. The final bar, in Figure 23, represents the global feeling that users had about the user modelling approach, concerning both the Personas and the user classification approach. The lowest value (0.57) is the classification given to the number of planned primary profiles. Although positive, it can create a feeling that the Personas are insufficient for real user needs. However, the idea concluded from the evaluation session and post discussion is that the number of profiles is adequate, and most people gave it a 0 classification not due to being a low number, but as a neutral value, because they were not surprised by them – they were expecting this approach.

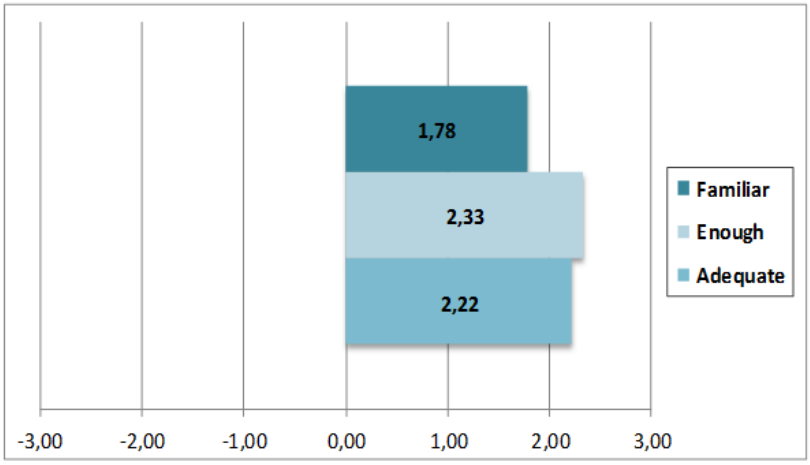

Figure 24 – ADSyS global opinion for scheduling problem definition

Figure 24 reveals the classification given to ADSyS in a global scope. A clear reading from the grades is that each participant, from the less experienced to the most, felt comfortable using the system, as it was familiar due to its dynamic adaptation to their experience. ADSyS is also sufficient and adequate for the user needs, from a beginner who needs guidance to an expert who desires advanced features. This classification also proves that the developed Personas are correct and symbolize truthfully the common users for ADSyS.

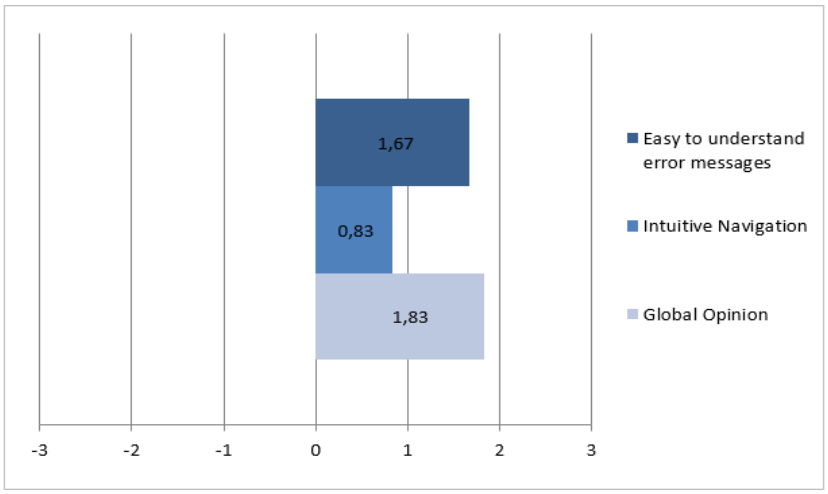

Figure 25 – MI adaptation scores

The most significant information from the second session is presented in Figure 25. As previously stated, the most relevant information from this session is to understand completely new users (to ADSyS or other scheduling systems) as they interact with the interface. The results clearly support the UM module, revealing positive results for the global adaptation. Specifically, displaying messages easy to understand and having an intuitive navigation is crucial for less experienced users.

Throughout the sessions, the users were encouraged to put forward suggestions for improvement on the ADSyS prototype. Several impressions were given, such has:

- The use of a more straightforward terminology for identifying the available mechanisms when selecting an algorithm;
- A clearer explanation on what mechanisms are automated;
- Enable window dragging on the multi-view feature;
- Implementation of shortcut keys for common tasks (such as copy, paste or to access other editors) and an overall enhanced use of the keyboard.

Most of the ideas can be correlated to the state of the ADSyS system, since it is a prototype (and not full concluded). These suggestions were all welcomed and were catalogued for future developments of ADSyS.

The clear result from both sessions is encouraging for the user modelling module development, as users from all profiles (beginner to expert) gave very positive classifications to its content. The overall system results prove that the system adaptation is successful, increasing the usability and user efficiency by employing the proposed user modelling architecture.

6.3 Summary

In order to understand the adequateness of the proposed user modelling architecture a study was performed, analysing the user classifiers (ANN and BN) from an accuracy standpoint and the system adaptation from a usability perspective.

Concerning the user classifiers, the conclusions were as expected from the previously presented literature review. Both the Bayesian and the Neural Networks are capable of performing user classification tasks with a similar accuracy level. A BN is faster to implement but using an ANN presents a larger potential to predict extreme user cases. The decision to select one structure over the other should be made based on the existence of a previous case base that can be used to train the ANN or a proper user study that can be utilised to define the BN CPD table.

On the subject of the system's adaptation, usability sessions were held with various users that had several levels of expertise. Users from all profiles gave very positive classifications to the system adaptation, which guarantees a higher usability when using the ADSyS system to all users.

Overall, the developed architecture can be deemed successfully, as it achieves the proposed objectives. The theoretic questions (which classifier to use) are answered and, perhaps with greater relevancy, the users have supported the developed techniques, being able to use ADSyS with a higher level of efficiency.

7 Conclusion

The central reason of this dissertation was to conceive an architecture for user modelling that could serve as a foundation for future adaptive systems. It is incorporated in ADSyS (a scheduling system) working as proof of concept so that it is usable not only in in scheduling but any type of system – if it is able to characterize users in such a complex environment it has the potential to work in any situation. This chapter presents an overall summary of the dissertation, resumes the most relevant contributions, underlines some limitations of the developed work and puts forward proposals for future improvements.

7.1 Summary

The principal motivation for this work emerged from the need for more effective and efficient methods to support new users in complex systems. Reducing the time required by a user to obtain the desired result, either via recommendations or by offering certain tools earlier, can be a difference maker. That is the incentive for adopting user modelling techniques: to increase system's usability. The higher it is the greater the effectiveness rates are, as users requires less time to accomplish specific task and are able to learn and understand how to operate the system.

The developed user modelling architecture was implemented on the ADSyS system, a high complexity dynamic system used to solve scheduling problems subject to dynamic circumstances.

This dissertation started with a literature review on the topic of Adaptive Hypermedia, the scientific "parent" of user modelling. Adaptive Hypermedia techniques can be used to increase the utility of a system. A study on how to adapt content on any system was put forward, including an apt description of the six fundamental Adaptive Hypermedia questions (Brusilovsky, 1996) and also containing a description of every component needed to create an Adaptive Hypermedia system.

Eventually, user modelling split up from the AH scientific area, becoming by itself a main subject of research. This work presented a history overview of the user modelling field, from its

inception to the most recent trends and developments. A thorough guide on how decide what type of information to maintain on a UM, how to initialize it and keep it up to date is presented. Then, an exposition of available user modelling techniques to classify users is put forward, were the Bayesian and Artificial Neural Networks are highlighted.

In order to understand the design decisions taken through the developed work it is necessary to known the system were the proposed architecture is applied. The ADSyS architecture is described in detail, including a brief review on relevant topics (such as Machine Learning or Dynamic Scheduling) and a comprehensive explanation of the components which are responsible for interacting with the users.

To adapt any system to its users, a proper system design should be planned. An architecture for user modelling is proposed as the main contribution of this dissertation. It is composed by a user information unit, a mathematical structure able to classify users and the technique to use when adapting content. A complete manual on how to construct and apply the developed architecture to any system is put forward. Regarding the user classifier two structures were selected: an ANN and a BN. A detailed guide on how to plan, prepare and train them to deal with user information has been provided.

One of the objectives of this work was to analyse and compare both structures to discover potential advantages of each approach. A computational study is presented in order to compare the approaches based on Bayesian and Artificial Neural Networks to model users' behaviour and profiles on ADSyS. As measured, Bayesian Networks are much simpler and can easily be update; however, they can only work when variables are independent and most of the definition has to be done by the developer, particularly during the CPD table definition. A BN should be advantageous when starting from scratch, with no previous work. An ANN is faster, easier and even the more correct method to implement if there is already a proper case base to perform the network training. However, there will always be exception to this theoretical rule, hence the need for a case by case analysis and the impossibility of stating one method as superior to the other.

At last, the results from the usability sessions were presented. They support the evaluation of the proposed classification and that the system adaptation is beneficial for system users. Two evaluation sessions were held with various users that had contrasting levels of expertise. Users from all profiles gave very positive classifications to the system adaptation, which guarantees that the developed architecture can be considered successfully, as it achieves the proposed objectives.

7.2 Main Contributions

The main contribution of this work was the definition of an architecture for user modelling, tested on a scheduling system and designed to work on other categories. The ADSyS system was equipped with the capacity to recognize its users, being able to know their level of knowledge. It was also prepared to adapt the system content and navigation according to their level of expertise, creating an easier system for new users and providing advanced tools to the more expertise.

Another relevant contribution was the identification of when using an ANN or a BN can be advantageous. During this work it was also presented a state of the art analysis on the topics of AH and User Modelling.

Other relevant contributions include a descriptive guide on how to conduct usability evaluation sessions to assess the state of a certain system and how to train and define the information for a Bayesian and an Artificial Neural Network.

Here also identified as contributions has been the creation and presentation of scientific papers related to the work developed in this dissertation. The scientific publications are:

- Cunha, B., Madureira, A. & Pereira, J.P. (2015). User modelling in scheduling system with Artificial Neural Networks. Information Systems and Technologies (CISTI), 2015 10th Iberian Conference on. pp. 1–6.
- Madureira, A., Cunha, B., Pereira, J.P., Gomes, S., Pereira, I., Jorge M. Santos & Abraham, A. (2014). Using Personas for Supporting User Modeling on Scheduling Systems. In: Proceedings of the International Conference on Hybrid Intelligent Systems. 2014, IEEE Press, pp. 279–284.
- Madureira, A., Cunha, B., Pereira, J.P., Pereira, I. & Gomes, S. (2014). An Architecture for User Modeling on Intelligent and Adaptive Scheduling Systems. In: Sixth World Congress on Nature and Biologically Inspired Computing (NaBIC). 2014, IEEE Press, pp. 103–108.
- Madureira, A., Gomes, S., Cunha, B., Pereira, J.P., Santos, J.M. & Pereira, I. (2014). Prototype of an Adaptive Decision Support System for Interactive Scheduling with MetaCognition and User Modeling Experience. In: Sixth World Congress on Nature and Biologically Inspired Computing (NaBIC). 2014, IEEE Press, pp. 145–150.

7.3 Limitations and future work

As the work presented in this dissertation was developed some limitations and vulnerabilities were identified. One of the limitations is related to the uniqueness and intricacy associated with every system that wants to adapt its content to system users: even if an architecture for user modelling is presented with the aspiration of it suiting any type of system, it has only been proved on a specific scheduling system. It is never possible to say that the proposed architecture will be the most appropriate in any type of system, as specific systems with certain restrictions could have better results with another approach.

Other vulnerability is the selection of user classifiers. To select the mathematical structure to classify users a proper study was conducted, comparing the various solutions. In the end, both the BN and ANN were selected to perform a deeper examination but the argument can be made that every mathematical possibility should have been considered – not only compare these two networks, but also other techniques (such as Decision Trees or stereotypes). This would obviously create a large amount of work but, without it, it is not possible to state the experienced structures as the most appropriate, effective considering the lack of significance.

At last, the training of the user classifiers is also subject of some vulnerability. To create the weights for the BN a rigorous process was followed (described on section 5.3.1). Nonetheless,

it is very difficult to be 100% sure that the made decisions were the correct ones (even with such respectable results) since it is not possible to test the classification accuracy for all possible combinations of weights. Regarding the ANN training, there will always exist the possibility that a particular set of combination of training and test cases would create a more accurate network. This issue is related to the unfeasibility of creating and testing Neural Networks for each possible combination. During the training and development of the user classifiers, the mind-set was never to create the perfect solution (which is always doubtful) but to create an adequate result, able to proper classify its users with satisfactory results.

With the purpose of amending the identified issues, it is suggested as future work the application of the proposed architecture on other types of systems, preferably on certain complex areas that benefit from content adaptation (such as medicine and education); the continuous assessment of the usability of the ADSyS system with future evaluation sessions with a distinct audience; a study on how the scale of the system and, specifically, the number of users would affect the performance of the user classifiers; and the development of a solution that would allow the cooperation between systems that use the proposed architecture in order to better identify its users, using ontologies or any other standardized approach for their communication. It is also noteworthy that, regarding the ADSyS system, all of the suggestions gathered on the evaluation sessions from the users (which are presented on section 6.2) are documented for future improvements.

References

Al-Omar, K. & Rigas, D. (2009). Comparison of adaptive, adaptable and mixed-initiative menus. In: *2009 International Conference on CyberWorlds, CW '09*. 2009, Bradford: IEEE, pp. 292–297.

Alonso, E., D'inverno, M., Kudenko, D., Luck, M. & Noble, J. (2001). Learning in multi-agent systems. *The Knowledge Engineering Review*. 16 (03). p.pp. 277–284.

Amato, F., López, A., Peña-Méndez, E.M., Vaňhara, P., Hampl, A., Havel, J., Lopez, A., Peña-Méndez, E.M., Vaňhara, P., Hampl, A. & Havel, J. (2013). Artificial neural networks in medical diagnosis. *Journal of applied biomedicine*. 11 (2). p.pp. 47–58.

Anthony, M. (1997). Artificial Neural Networks. In: L. B. and R. Wilson (ed.). *Graph Connections*. Oxford University Press, pp. 247–260.

Ardissono, L., Goy, A. & Petrone, G. (2008). A Framework for the Development of Distributed, Context-Aware Adaptive Hypermedia Applications. In: W. Nejdl, J. Kay, P. Pu, & E. Herder (eds.). *Adaptive Hypermedia and Adaptive Web-Based Systems SE - 29*. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 259–262.

Armstrong, R., Freitag, D. & Joachims, T. (1995). Webwatcher: A learning apprentice for the world wide web. In: *AAAI Spring symposium on Information gathering from Heterogeneous, distributed environments*. 1995, AAAI Press, pp. 6–12.

Aroyo, L., Stash, N., Wang, Y., Gorgels, P. & Rutledge, L. (2007). CHIP Demonstrator: Semantics-Driven Recommendations and Museum Tour Generation. In: K. Aberer, K.-S. Choi, N. Noy, D. Allemang, K.-I. Lee, L. Nixon, J. Golbeck, P. Mika, D. Maynard, R. Mizoguchi, G. Schreiber, & P. Cudré-Mauroux (eds.). *The Semantic Web SE - 64*. Lecture Notes in Computer Science. 2007, Springer Berlin Heidelberg, pp. 879–886.

Ayodele, T., Zhou, S. & Khusainov, R. (2009). Evolving email clustering method for email grouping: A machine learning approach. In: *2009 Second International Conference on the Applications of Digital Information and Web Technologies*. August 2009, IEEE, pp. 357–362.

Baclawski, K.P. (2004). Bayesian network development. In: *New Trends in Software Methodologies, Tools, and Techniques*. IOS Press, pp. 18–48.

Blomkvist, S. (2002). The User as a personality. In: *Using Personas as a tool for design. Position paper for the course workshop 'Theoretical perspectives in Human-Computer Interaction' at IPLab, KTH*. 2002.

De Bra, P., Houben, G.-J. & Wu, H. (1999). AHAM : A Dexter-based Reference Model for Adaptive Hypermedia. In: Springer (ed.). *Proceedings of the ACM Conference on Hypertext and Hypermedia*. 1999, pp. 147–156.

De Bra, P., Pechenizkiy, M., Van Der Sluijs, K. & Smits, D. (2008). GRAPPLE: Integrating adaptive learning into learning management systems. In: Joseph Luca & E. R. Weippl (eds.). *World Conference on Educational Media and Technology*. 2008, Vienna, Austria: AACE, pp. 5183–5188.

De Bra, P., Smits, D. & Stash, N. (2006). The design of AHA! In: *Proceedings of the seventeenth conference on Hypertext and hypermedia*. 2006, Odense, Denmark: ACM, pp. 133–134.

Bransford, J.D., Brown, A.L. & Cocking, R.R. (1999). *How people learn: Brain, mind, experience, and school.* Washington DC: National Academy Press.

Breiman, L. (2001). Statistical Modeling: The Two Cultures. *Statistical Science*. 16 (3). p.pp. 199–231.

Brigham, T.J. (2013). Personas: stepping into the shoes of the library user. *Medical reference services quarterly*. 32. p.pp. 443–50.

Brusilovsky, P. (2001). Adaptive hypermedia. *User modeling and user-adapted interaction*. 11 (1-2). p.pp. 87–110.

Brusilovsky, P. (2000). Adaptive Hypermedia: From Intelligent Tutoring Systems to Web-Based Education. *Intelligent Tutoring Systems*. 1839. p.pp. 1–7.

Brusilovsky, P. (2007). Adaptive Navigation Support. In: P. Brusilovsky, A. Kobsa, & W. Nejdl (eds.). *The Adaptive Web*. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 263–290.

Brusilovsky, P. (2008). Adaptive navigation support for open corpus hypermedia systems. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. 2008, pp. 6–8.

Brusilovsky, P. (1996). Methods and techniques of adaptive hypermedia. *User Modeling and User-Adapted Interaction*. 6 (2-3). p.pp. 87–129.

Brusilovsky, P. (1994). The Construction and Application of Student Models in Intelligent Tutoring Systems. *The construction and application of student models in intelligent tutoring systems*. 32. p.pp. 70–89.

Brusilovsky, P., Eklund, J. & Schwarz, E. (1998). Web-based education for all: A tool for developing adaptive courseware. In: *Computer Networks and ISDN Systems*. 1998, pp. 291–300.

Brusilovsky, P. & Henze, N. (2007). Open Corpus Adaptive Educational Hypermedia. In: P. Brusilovsky, A. Kobsa, & W. Nejdl (eds.). *The Adaptive Web: Methods and Strategies of Web Personalization*. Springer-Verlag Berlin Heidelberg, pp. 671–696.

Brusilovsky, P. & Pesin, L. (1998). Adaptive navigation support in educational hypermedia: An evaluation of the ISIS-Tutor. *Journal of computing and Information Technology*. 6 (1). p.pp. 27–38.

Brusilovsky, P. & Pesin, L. (1995). Visual annotation of links in adaptive hypermedia. In: *Conference companion on Human factors in computing systems - CHI '95*. 7 May 1995, New York, New York, USA: ACM Press, pp. 222–223.

Carbonell, J.G., Michalski, R.S. & Mitchell, T.M. (1983). *Machine Learning: An Artificial Intelligence Approach*. CMU-CS. Los Altos, CA: Morgan Kaufmann Publishers Inc.

Carr, B. & Goldstein, I. (1977). *Overlays: A theory of modelling for computer aided instruction*. Massachusetts Institute of Technology.

Carro, R., Ortigosa, A., Martín, E. & Schlichter, J. (2003). Dynamic Generation of Adaptive Web-Based Collaborative Courses. In: J. Favela & D. Decouchant (eds.). *Groupware: Design, Implementation, and Use SE - 17*. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 191–198.

Challis, D. (2005). Committing to quality learning through adaptive online assessment. *Assessment & Evaluation in Higher Education*. 30 p.pp. 519–527.

Chittaro, L. & Ration, R. (2000). Adding adaptive features to virtual reality interfaces for e-commerce. In: *Adaptive Hypermedia and Adaptive Web-Based Systems*. 2000, Springer, pp. 86–97.

Cohen, P.R. & Perrault, C.R. (1979). Elements of a Plan-Based Theory of Speech Acts. *Cognitive Science*. 3 (3). p.pp. 177–212.

Conlan, O., Dagger, D. & Wade, V. (2002). Towards a Standards-based Approach to e-Learning Personalization using Reusable Learning Objects. In: *Proceedings*. 2002, pp. 210–217.

Conlan, O., Wade, V., Bruen, C. & Gargan, M. (2006). Multi-model, metadata driven approach to adaptive hypermedia services for personalized elearning. In: *Adaptive Hypermedia and Adaptive Web-Based Systems*. 2006, pp. 100–111.

Cooper, A. (1999). *The Inmates Are Running the Asylum*. Indianapolis, USA: Macmillan Publishing Co., Inc.

Cunha, B., Madureira, A. & Pereira, J.P. (2015). User modelling in scheduling system with artificial neural networks. *Information Systems and Technologies (CISTI), 2015 10th Iberian Conference on*. p.pp. 1–6.

Dayhoff, J.E. & DeLeo, J.M. (2001). Artificial neural networks: opening the black box. *Cancer*. 91 (8 Suppl). p.pp. 1615–35.

Dey, A.K. & Abowd, G.D. (1999). Towards a Better Understanding of Context and Context-Awareness. *Computing Systems*. 40. p.pp. 304–307.

Dhar, V. (2013). Data Science and Prediction. *Commun. ACM*. 56 (12). p.pp. 64–73.

Edwards, D.M. & Hardman, L. (1989). 'Lost in Hyperspace': Cognitive mapping and navigation in a hypertext environment. *Hypertext: Theory into practice*. p.pp. 90 – 105.

Eklund, J. & Brusilovsky, P. (1998). The value of adaptivity in hypermedia learning environments: A short review of empirical evidence. In: *Proceedings of the the Ninth ACM Conference on Hypertext and Hypermedia*. 1998, pp. 11–17.

Fahlman, S.E. & Lebiere, C. (1990). The cascade-correlation learning architecture. In: *Advances in neural information processing systems 2*. San Francisco, USA: Morgan Kaufmann Publishers Inc., pp. 524–532.

Farzan, R. & Brusilovsky, P. (2006). Social Navigation Support in a Course Recommendation System. In: V. Wade, H. Ashman, & B. Smyth (eds.). *Adaptive Hypermedia and Adaptive Web-Based Systems SE - 11*. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 91–100.

Fernandes, N., Kaklanis, N., Votis, K., Tzovaras, D. & Carriço, L. (2014). An analysis of personalized web accessibility. In: *Proceedings of the 11th Web for All Conference on - W4A '14*. 7 April 2014, New York, New York, USA: ACM Press, pp. 1–10.

Finin, T. (1989). GUMS — A General User Modeling Shell. In: A. Kobsa & W. Wahlster (eds.). *User Models in Dialog Systems SE - 15*. Symbolic Computation. Springer Berlin Heidelberg, pp. 411–430.

Finin, T. & Orager, D. (1986). GUMS: A General User Modeling System. *Proceedings of the workshop on Strategic computing natural language of the Human Language Technology Conference*. p.pp. 224–230.

Frias-Martinez, E., Chen, S.Y. & Liu, X. (2006). Survey of data mining approaches to user modeling for adaptive hypermedia. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*. 36 (6). p.pp. 734–749.

Froschl, C. (2005). User modeling and user profiling in adaptive e-learning systems. *Graz, Austria: Master Thesis*.

Goren, S., Sabuncuoglu, I. & Koc, U. (2012). Optimization of schedule stability and efficiency under processing time variability and random machine breakdowns in a job shop environment. *Naval Research Logistics*. 59. p.pp. 26–38.

Han, B. (2001). *Student Modelling and Adaptivity in web based Learning Systems*. Master's t. M. University (ed.). New Zealand.

Hanisch, F., Muckenhaupt, M., Kurfess, F. & Straßer, W. (2006). The AHES taxonomy: Extending adaptive hypermedia to software components. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. 2006, pp. 342–345.

Hearst, M.A. (1999). Trends & Controversies: Mixed-initiative interaction. *IEEE Intelligent Systems*. 14. p.pp. 14–23.

Heckmann, D. & Krueger, A. (2003). A User Modeling Markup Language (UserML) for Ubiquitous Computing. *User Modeling 2003*. p.pp. 393–397.

Heckmann, D., Schwartz, T., Brandherm, B., Schmitz, M. & von Wilamowitz-Moellendorff, M. (2005). Gumo - The General User Model Ontology. In: *Proceedings of UM 2005: 10th International Conference on User modeling*. 2005, pp. 428–432.

Van Heerden, B., Aldrich, C. & Du Plessis, A. (2008). Predicting student performance using artificial neural network analysis. *Medical Education*. 42 (5). p.pp. 516–517.

Hendrix, M. & Cristea, A. (2008). Meta-levels of Adaptation in Education. In: *11th IASTED International Conference on Computers and Advanced Technology in Education*. 1 January 2008, V. Uskov (Ed.), IASTED.

Henze, N. (2000). Adaptive hyperbooks: Adaptation for project-based learning resources. *University of Hannover: PhD thesis*.

Henze, N. & Nejdl, W. (2004). A logical characterization of adaptive educational hypermedia. *New Review of Hypermedia and Multimedia*. 10 p.pp. 77–113.

Hix, D. & Hartson, H.R. (1993). *Developing User Interfaces: Ensuring Usability Through Product & Process*. New York, NY, USA: John Wiley & Sons, Inc.

Horvitz, E. (1999). Principles of mixed-initiative user interfaces. In: *CHI '99 Proceedings of the SIGCHI conference on Human Factors in Computing Systems*. 1999, Chicago: ACM New York, pp. 159–166.

James, G., Witten, D., Hastie, T. & Tibshirani, R. (2013). *An Introduction to Statistical Learning*. Springer Texts in Statistics. New York, NY: Springer New York.

Jennings, A. & Higuchi, H. (1993). A user model neural network for a personal news service. *User Modeling and User-Adapted Interaction*. 3. p.pp. 1–25.

Kadie, C., Hovel, D. & Horvitz, E. (2001). MSBNx: A component-centric toolkit for modeling and inference with Bayesian networks. *WA, Technical Report MSR-TR-2001*. (July).

Kaplan, C., Fenwick, J. & Chen, J. (1993). Adaptive hypertext navigation based on user goals and context. *User Modeling and User-Adapted Interaction*. 3 (3). p.pp. 193–220.

Karray, F., Alemzadeh, M., Abou, J. & Arab, S.M.N. (2008). Human-Computer Interaction: Overview on State of the Art. *The International Journal on Smart Sensing and Intelligent Systems*. 1 (1).

Kay, J. (2000). Stereotypes, student models and scrutability. In: G. Gauthier, C. Frasson, & K. VanLehn (eds.). *Intelligent Tutoring Systems*. 2000, Springer Berlin Heidelberg, pp. 19–30.

Kay, J. & McCalla, G. (2012). Coming of age: celebrating a quarter century of user modeling and personalization: Guest editors' introduction. *User Modeling and User-Adapted Interaction*. 22 (1-2). p.pp. 1–7.

Kinshuk (1996). Computer Aided Learning for Entry Level Accountancy Students. *De Montfort University: PhD Thesis*.

Knutov, E., De Bra, P. & Pechenizkiy, M. (2009). AH 12 years later: a comprehensive survey of adaptive hypermedia methods and techniques. *New Review of Hypermedia and Multimedia*. 15 p.pp. 5–38.

Kobsa, A. (2001). Generic User Modeling Systems. *User Modeling and User-Adapted Interaction*. 11 (1-2). p.pp. 49–63.

Kobsa, A. (1990). Modeling the user's conceptual knowledge in BGP-MS, a user modeling shell system. *Computational Intelligence*. 6. p.pp. 193–208.

Kobsa, A. (2007). Privacy-Enhanced Web Personalization. *Communications of the ACM*. 50 (8). p.pp. 628–670.

Kobsa, A. (1993). User Modeling : Recent Work , Prospects and Hazards 1. In: *Adaptive User Interfaces: Principles and Practice*. pp. 111–128.

Koch, N. & Wirsing, M. (2002). The Munich Reference Model for Adaptive Hypermedia Applications. In: *Adaptive Hypermedia and Adaptive Web-Based Systems*. 2002, pp. 213–222.

Korb, K.B. & Nicholson, A.E. (2010). *Bayesian artificial intelligence*. 2nd Ed. Boca Raton, FL, USA: CRC Press, Inc.

Krenker, A., Kos, A. & Bešter, J. (2011). *Introduction to the Artificial Neural Networks*. Ljubljana, Slovenia: INTECH Open Access Publisher.

Kurhila, J., Miettinen, M., Nokelainen, P. & Tirri, H. (2002). Educo - A Collaborative Learning Environment Based on Social Navigation. In: P. De Bra, P. Brusilovsky, & R. Conejo (eds.). *Adaptive Hypermedia and Adaptive Web-Based Systems SE - 26*. Lecture Notes in Computer Science. Málaga, Spain: Springer Berlin Heidelberg, pp. 242–252.

De La Passardiere, B. & Dufresne, A. (1992). Adaptive navigational tools for educational hypermedia. In: I. Tomek (ed.). *Computer Assisted Learning SE - 48*. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 555–567.

Lee, Y.H., Kumara, S.R.T. & Chatterjee, K. (2003). Multiagent based dynamic resource scheduling for distributed multiple projects using a market mechanism. *Journal of Intelligent Manufacturing*. 14. p.pp. 471–484.

León, H.B., Rego, H., Moreira, T. & Peñalvo, F.G. (2005). A model for online assessment in adaptive e-learning platform. In: A. Méndez-Vilas, B. González-Pereira, J. M. González, & J. A. M. González (eds.). *Recent Research Developments In Learning Technologies*. Badajoz, Spain: Formatex, pp. 1–5.

Levacher, K., Lawless, S. & Wade, V. (2012). Slicepedia: providing customized reuse of open-web resources for adaptive hypermedia. In: *HT '12 Proceedings of the 23rd ACM conference on Hypertext and social media*. 2012, Milwaukee, USA: ACM New York, pp. 23–32.

Li, X. & Ji, Q. (2005). Active affective state detection and user assistance with dynamic Bayesian networks. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*. 35 (1). p.pp. 93–105.

Madureira, A. (2010). Técnicas Emergentes de Optimização no Suporte à Tomada de Decisão. *Technical report: Instituto Superior de Engenharia do Instituto Politécnico do Porto*.

Madureira, A., Cunha, B., Pereira, J.P., Gomes, S., Pereira, I., Jorge M. Santos & Abraham, A. (2014a). Using Personas for Supporting User Modeling on Scheduling Systems. In: *Proceedings of the International Conference on Hybrid Intelligent Systems*. 2014, IEEE Press, pp. 279–284.

Madureira, A., Cunha, B., Pereira, J.P., Pereira, I. & Gomes, S. (2014b). An Architecture for User Modeling on Intelligent and Adaptive Scheduling Systems. In: *Sixth World Congress on Nature and Biologically Inspired Computing (NaBIC)*. 2014.

Madureira, A., Gomes, N., Santos, J. & Ramos, C. (2007a). Cooperation Mechanism for Team-Work based Multi-Agent System in Dynamic Scheduling through Meta-Heuristics. *2007 IEEE International Symposium on Assembly and Manufacturing*.

Madureira, A., Gomes, S., Cunha, B., Pereira, J.P., Santos, J.M. & Pereira, I. (2014c). Prototype of an Adaptive Decision Support System for Interactive Scheduling with MetaCognition and User Modeling Experience. In: *Sixth World Congress on Nature and Biologically Inspired Computing (NaBIC)*. 2014.

Madureira, A. & Pereira, I. (2010). Intelligent Bio-Inspired system for manufacturing scheduling under uncertainties. In: *2010 10th International Conference on Hybrid Intelligent Systems, HIS 2010*. 2010, pp. 109–112.

Madureira, A., Pereira, I. & Falcão, D. (2013). Dynamic Adaptation for Scheduling Under Rush Manufacturing Orders With Case-Based Reasoning. In: *Int. Conf. on Algebraic and Symbolic Computation*. 2013.

Madureira, A., Ramos, C. & Silva, S.D. do C. (2002). A coordination mechanism for real world scheduling problems using genetic algorithms. *Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No.02TH8600)*. 1.

Madureira, A., Santos, J. & Gomes, N. (2007b). Hybrid multi-agent system for cooperative dynamic scheduling through meta-heuristics. In: *Proceedings of The 7th International Conference on Intelligent Systems Design and Applications, ISDA 2007*. 2007, pp. 9–14.

Madureira, A., Santos, J.M., Gomes, S., Cunha, B., Pereira, J.P. & Pereira, I. (2014d). Manufacturing Rush Orders Rescheduling: a Supervised Learning Approach. In: *Sixth World Congress on Nature and Biologically Inspired Computing (NaBIC)*. 2014.

Martín, E., Carro, R. & Rodríguez, P. (2006). A mechanism to support context-based adaptation in M-Learning. *Innovative Approaches for Learning and Knowledge Sharing*. p.pp. 302–315.

Mei, L. & Easterbrook, S. (2007). Evaluating user-centric adaptation with goal models. In: *Proceedings - ICSE 2007 Workshops: First International Workshop on Software Engineering for Pervasive Computing Applications, Systems, and Environments, SEPCASE'07*. 2007.

Michie, E.D., Spiegelhalter, D.J. & Taylor, C.C. (1994). Machine Learning , Neural and Statistical Classification. *Technometrics*. 37. p.p. 459.

Mitchell, T.M. (1997). *Machine Learning*. New York, NY, USA: McGraw-Hill, Inc.

Mohamad, Y. & Kouroupetroglou, C. (2014). Research Report on User Modeling for Accessibility. *W3C WAI Research and Development Working Group (RDWG)*. (1).

Monostori, L., Váncza, J. & Kumara, S.R.T. (2006). Agent-Based Systems for Manufacturing. *CIRP Annals - Manufacturing Technology*. 55 p.pp. 697–720.

Motti, V.G. & Vanderdonckt, J. (2013). A computational framework for context-aware adaptation of user interfaces. In: *IEEE 7th International Conference on Research Challenges in Information Science (RCIS)*. May 2013, IEEE, pp. 1–12.

Natarajan, S., Bui, H.H., Tadepalli, P., Kersting, K. & Wong, W.-K. (2008). Logical Hierarchical Hidden Markov Models for Modeling User Activities. *Lecture Notes In Artificial Intelligence; Vol. 5194*.

Nikovski, D. (2000). Constructing Bayesian networks for medical diagnosis from incomplete and partially correct statistics. *Knowledge and Data Engineering, IEEE Transactions on*. 12 (4). p.pp. 509–516.

Nwana, H.S. (1996). Software agents: An overview. *The knowledge engineering review*. 11 (03). p.pp. 205–244.

O'Connor, E., Smeaton, A.F., O'Connor, N.E. & Regan, F. (2012). A Neural Network Approach to Smarter Sensor Networks for Water Quality Monitoring. *Sensors*. 12 (4). p.pp. 4605–4632.

Oberlander, J. & O'Donnell, M. (1998). Conversation in the museum: experiments in dynamic hypermedia with the intelligent labelling explorer. *New Review of Hypermedia and MultimediaReview of Hypermedia*. 4. p.pp. 11–32.

Ohene-Djan, J., Bailey, C.P., Wills, G.B. & C, H. (2003). Is it possible to devise a unifying model of adaptive hypermedia and is one necessary? In: *14th Conference on Hypertext and Hypermedia*. 1 January 2003, Nottingham, UK.

Ohene-Djan, J. & Fernandes, A.A. (2002). Modelling personalisable hypermedia: The Goldsmiths Model. *New Review of Hypermedia and Multimedia*. 8 (1). p.pp. 99–137.

Ouelhadj, D., Cowling, P.I. & Petrovic, S. (2003). Utility and stability measures for agent-based dynamic scheduling of steel continuous casting. *2003 IEEE International Conference on Robotics and Automation (Cat. No.03CH37422)*. 1.

Pazzani, M., Muramatsu, J. & Billsus, D. (1996). Syskill & Webert: Identifying interesting web sites. In: *Proceedings of the thirteenth national conference on Artificial intelligence - Volume 1*. 1996, pp. 54–61.

Piairo, J., Madureira, A., Pereira, J. & Pereira, I. (2013). A User-Centered Interface for Scheduling Problem Definition. In: Á. Rocha, A. M. Correia, T. Wilson, & K. A. Stroetmann (eds.). *Advances in Information Systems and Technologies SE - 101*. Advances in Intelligent Systems and Computing. Springer Berlin Heidelberg, pp. 1063–1073.

Pinedo, M.L. (2008). *Scheduling: Theory, algorithms, and systems*. 4th Ed. New York: Springer-Verlag New York.

Popescu, E. (2010). Adaptation provisioning with respect to learning styles in a Web-based educational system: an experimental study. *Journal of Computer Assisted Learning*. 26 (4). p.pp. 243–257.

Pruitt, J. & Grudin, J. (2003). Personas: practice and theory. In: *Proceedings of the 2003 conference on Designing for user experiences*. 2003, ACM, pp. 1–15.

Pruitt, J.S. & Adlin, T. (2006). Persona conception and gestation. In: *The Persona Lifecycle*. Morgan Kaufmann Publishers Inc., pp. 162–271.

Ramos, C. (2001). Sistemas de Apoio à Decisão com Inteligência Escalável. *Report: DEEC – Faculdade de Engenharia da Universidade do Porto*.

Raskutti, B., Beitz, A. & Ward, B. (1997). A feature-based approach to recommending selections based on past preferences. *User Modeling and User-Adapted Interaction*. 7. p.pp. 179–218.

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P. & Riedl, J. (1994). GroupLens : An Open Architecture for Collaborative Filtering of Netnews. In: *Proceedings of the 1994 ACM conference on Computer supported cooperative work*. 1994, pp. 175–186.

Rich, E. (1979a). Building and exploiting user models. In: *IJCAI'79 Proceedings of the 6th international joint conference on Artificial intelligence - Volume 2*. 1979, Morgan Kaufmann Publishers Inc., pp. 720–722.

Rich, E. (1979b). User Modeling via Stereotypes. *Cognitive Science*. 3. p.pp. 329–354.

Riedmiller, M. & Braun, H. (1992). RPROP - A Fast Adaptive Learning Algorithm. In: *Proceedings of the International Symposium on Computer and Information Science VII*. 1992.

Romero, C., Ventura, S., Bra, P. De & Castro, C. De (2003). Discovering Prediction Rules in AHA ! Courses. In: *9th International User Modeling Conference*. 2003, pp. 1–10.

Rutledge, L., Stash, N., Wang, Y. & Aroyo, L. (2008). Accuracy in Rating and Recommending Item Features. In: W. Nejdl, J. Kay, P. Pu, & E. Herder (eds.). *Adaptive Hypermedia and Adaptive Web-Based Systems SE - 19*. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 163–172.

Salton, G. & McGill, M.J. (1983). *Introduction to modern information retrieval.* McGraw-Hill, Inc. New York.

Schein, A.I., Popescul, A., Ungar, L.H. & Pennock, D.M. (2002). Methods and metrics for cold-start recommendations. In: *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '02*. 11 August 2002, New York, New York, USA: ACM Press, p. 253.

Self, J. (1994). Formal Approaches to Student Modelling. In: J. Greer & G. McCalla (eds.). *Student Modelling: The Key to Individualized Knowledge-Based Instruction SE - 12*. NATO ASI Series. Springer Berlin Heidelberg, pp. 295–352.

Self, J. (1993). Model-based cognitive diagnosis. *User Modeling and User-Adapted Interaction*. 3 (1). p.pp. 89–106.

Shneiderman, B. (1998). *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. 1st Ed. Reading, MA: Addison-Wesley.

Smits, D. & De Bra, P. (2011). GALE: A Highly Extensible Adaptive Hypermedia Engine. In: *HT 2011 - Proceedings of the 22nd ACM Conference on Hypertext and Hypermedia (Eindhoven, The Netherlands, June 6-9, 2011)*. 2011, pp. 63–72.

Sosnovsky, S. & Dicheva, D. (2010). Ontological technologies for user modelling. *International Journal of Metadata, Semantics and Ontologies*. 5 (1). p.p. 32.

Specht, M., Kravcik, M., Klemke, R., Pesin, L. & Huttenhain, R. (2002). Adaptive Learning Environment for Teaching and Learning in WINDS. *LECTURE NOTES IN COMPUTER SCIENCE*. p.pp. 572–575.

Stergiou, C. & Siganos, D. (2011). Neural Networks. *Report: Imperial College London*.

Tintarev, N. & Masthoff, J. (2008). The Effectiveness of Personalized Movie Explanations: An Experiment Using Commercial Meta-data. In: W. Nejdl, J. Kay, P. Pu, & E. Herder (eds.). *Adaptive Hypermedia and Adaptive Web-Based Systems SE - 23*. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 204–213.

Triantafillou, E., Pomportsis, A., Demetriadis, S. & Georgiadou, E. (2004). The value of adaptivity based on cognitive style: an empirical study. *British Journal of Educational Technology*. 35. p.pp. 95–106.

Tsiriga, V. & Virvou, M. (2003). Initializing student models in Web-based ITSs: a generic approach. In: *Proceedings 3rd IEEE International Conference on Advanced Technologies*. 2003, IEEE Comput. Soc, pp. 42–46.

Varela, M.L.R., Aparício, J.N. & Silva, S.C. (2003). A web-based application for manufacturing scheduling. In: *Proceedings of the IASTED International Conference on Intelligent Systems and Control*. 2003, pp. 400–405.

Vassileva, J., McCalla, G. & Greer, J. (2003). Multi-agent multi-user modeling in I-Help. *User Modelling and User-Adapted Interaction*. 13. p.pp. 179–210.

Virzi, R.A. (1992). Refining the Test Phase of Usability Evaluation : How Many Subjects Is Enough ? *Human Error and Clinical Systems*. 34 (4). p.pp. 457–468.

Wang, Y. & Kobsa, A. (2013). A PLA-based privacy-enhancing user modeling framework and its evaluation. *User Modeling and User-Adapted Interaction*. 23 (1). p.pp. 41–82.

Webb, G. (1998). Preface to UMUAI Special Issue on Machine Learning for User Modeling. *User Modeling and User-Adapted Interaction*. 8 (1-2). p.pp. 1–3.

Webb, G.I., Pazzani, M.J. & Billsus, D. (2001). Machine Learning for User Modeling. *User Modeling and User-Adapted Interaction*. 11 (1-2). p.pp. 19–29.

Weber, G. & Specht, M. (1997). User Modeling and Adaptive Navigation Support in WWW-Based Tutoring Systems. In: *Proceedings of the Sixth International Conference UM97 Chia Laguna, Sardinia, Italy June 2–5 1997*. 1997, pp. 289–300.

Yan, T.W., Jacobsen, M., Garcia-Molina, H. & Dayal, U. (1996). From user access patterns to dynamic hypertext linking. *Computer Networks and ISDN Systems*. 28 p.pp. 1007–1014.

Zhang, H., Song, Y. & Song, H.-T. (2007). Construction of ontology-based user model for web personalization. In: *User Modeling 2007*. Springer, pp. 67–76.

Zukerman, I. & Albrecht, D.W. (2001). Predictive statistical models for user modeling. *User Modeling and User-Adapted Interaction*. 11. p.pp. 5–18.

Appendix

This appendix contains the guide for the usability evaluation sessions that were conducted in the ambit of this thesis. The presented version is the one used on the second session (detailed in 6.2, page 59) and is in Portuguese due to it being the language on which it was conducted.

Guião da sessão para a avaliação da usabilidade do protótipo ADSyS

Sessão para a avaliação da usabilidade
Questionário inicial
Teste de eficiência
Questionário após a experimentação do protótipo ADSyS

ADSyS - Sistema de Apoio à Decisão para Escalonamento

Sessão para a avaliação da usabilidade

Obrigado por ter aceitado participar nesta experiência. O seu objectivo principal consiste na avaliação das ideias subjacentes ao desenvolvimento do sistema ADSyS, um sistema de apoio à decisão adaptativo e interativo vocacionado para a área de escalonamento. O roteiro desta sessão está descrito na Tabela 1, indicando-se os tempos estimados para a duração de cada uma das tarefas previstas, num total esperado de cerca de 60 minutos.

Tabela 1: Roteiro da sessão de avaliação

1	Recepção (Descrição dos trabalhos)	5 min
2	Questionário inicial (Opinião e experiência sobre sistemas de escalonamento)	5 min
3	Apresentação do protótipo ADSyS, seguida de discussão	15 min
4	Realização do teste de eficiência	25 min
5	Questionário sobre o protótipo ADSyS (Após a realização do teste de eficiência)	10 min

A sessão será conduzida pelo Eng.º Bruno Cunha, o qual estará ao vosso dispor para todo e qualquer esclarecimento.

Os questionários apresentam uma escala de valores entre dois atributos opostos, sendo o valor médio (0) um valor neutro. O valor NA (*Não se Aplica*) corresponde à inexistência de resposta para a questão em causa.

Pretende-se averiguar a utilidade e a facilidade de utilização do sistema ADSyS, pelo que todos os comentários e sugestões serão bem-vindos. Durante a execução das tarefas "pense em voz alta". Não há respostas certas ou erradas. Não se sinta inibido para apontar aspectos negativos ou positivos, para enunciar expectativas frustradas ou recompensadas.

Para finalizar gostaria de lhe agradecer o tempo e o esforço despendidos.

ADSyS - Sistema de Apoio à Decisão para Escalonamento

Questionário inicial

1	**No âmbito do processo de escalonamento, considera os computadores** dispensáveis indispensáveis |_|_|_|0|_|_|_|
4	**Que instrumento prefere para realizar o processo de escalonamento** manualmente sistema computacional de escalonamento |_|_|_|0|_|_|_| NA |_|
	Como considera a utilidade dos sistemas de escalonamento no que se refere
5	• à afetação de recursos dispensáveis indispensáveis |_|_|_|0|_|_|_|
6	• à distribuição de tarefas dispensáveis indispensáveis |_|_|_|0|_|_|_|
7	• à programação e controlo da produção dispensáveis indispensáveis |_|_|_|0|_|_|_|
	Considera a utilização de sistemas de escalonamento
8	• na afetação de recursos muito difícil muito fácil |_|_|_|0|_|_|_| NA |_|
9	• na distribuição de tarefas muito difícil muito fácil |_|_|_|0|_|_|_| NA |_|
10	• na programação e controlo da produção muito difícil muito fácil |_|_|_|0|_|_|_| NA |_|
	Na sua opinião um sistema de escalonamento serve para
11	• otimizar o processo de produção mal muito bem |_|_|_|0|_|_|_| NA |_|
12	• reduzir tempos mortos mal muito bem |_|_|_|0|_|_|_| NA |_|
13	• maximizar a utilização dos recursos mal muito bem |_|_|_|0|_|_|_| NA |_|

	Qual a sua apreciação sobre os melhores sistemas de escalonamento que conhece
14	• funcionalidade péssima excelente |_|_|_|0|_|_|_| NA |_|
15	• interacção péssima excelente |_|_|_|0|_|_|_| NA |_|
16	• manuais e outros auxílios péssima excelente |_|_|_|0|_|_|_| NA |_|
17	• opinião global péssima excelente |_|_|_|0|_|_|_| NA |_|
	Refira os sistemas de escalonamento que já utilizou
18	_____ _____ _____ _____ _____ _____ _____ _____ _____ _____ _____ _____ _____ _____ _____ _____

Data __/__/____

Participante n.º __

ADSyS - Sistema de Apoio à Decisão para Escalonamento

Teste de eficiência

O problema de escalonamento definido no ficheiro *A7 – FJSSP 8x5* é composto por um conjunto de oito tarefas a serem processadas em cinco máquinas. Cada uma das tarefas possui uma prioridade, que funciona como fator de penalização em caso de atraso na entrega, uma data de lançamento, uma data de entrega, e uma quantidade que corresponde ao número de itens a serem produzidos.

Por favor queira proceder à realização das seguintes tarefas. Não hesite em fazer comentários em voz alta, nem em pedir ajuda sempre que entender necessário. A realização da atividade deve ser iniciada apenas depois de receber indicações nesse sentido.

1. Gere e visualize o plano de escalonamento para este *order set* com a meta-heurística *Simulated Annealing*, solução inicial EDD e função objetivo CMax.
2. No plano de escalonamento resultante:
    a. Altere para 55 o tempo de processamento inicial da operação O4 da tarefa T8.
    b. Altere para 5 o tempo de processamento inicial da operação O1 da tarefa T3.
3. A tarefa T5 contém cinco operações, altere para 6 o tempo de processamento da operação O4.
4. Altere para 8 a data de lançamento da tarefa T7.
5. O valor máximo do horizonte temporal para este *order set* está definido com o valor 53, altere-o para 60.
6. Gere um novo plano de escalonamento (de modo similar ao ponto 1), de modo a visualizar a consequência das alterações efetuadas.
7. Compare os dois planos, em simultâneo, usando a funcionalidade *Multiview*.
8. Elimine a operação O5 da tarefa T1.
9. Altere para 9 o tempo de processamento da operação O2 da tarefa T3.
10. A operação O1 da tarefa T6 é processada pela máquina M1 altere-a para a máquina M4.
11. Crie uma nova tarefa (T9) com as seguintes características:
    Tarefa: T9
    Descrição: T9
    Data de lançamento: 3
    Data de Entrega: 47
    Prioridade: 1
    Quantidade: 1

Tarefa	ID op.	Descrição da operação	Máq.	Tempo de proc.	Op. precedentes	Op. seguintes
T9 Moldura 10x15	O1	Cortar ripas de madeira	M3	6	-	O2
	O2	Limar ripas	M5	4	O1	O3
	O3	Montar a moldura de madeira	M4	4	O2	O4
	O4	Colocar suportes para pendurar e segurar	M4	3	O3	O6
	O5	Cortar vidro	M2	6	-	O6
	O6	Montar vidro na moldura de madeira	M4	2	O5, O4	-

12. Adicione uma nova operação à tarefa T3 com as seguintes características:

    ID operação: O6
    Descrição da operação: O6
    Máquina: M2
    Tempo de processamento: 5
    Operações precedentes: O5
    Operações seguintes: -

13. Gere um novo plano de escalonamento (de modo similar ao ponto 1), de modo a visualizar as alterações efetuadas.
14. No novo plano de escalonamento simule a chegada de uma nova tarefa.
15. Compare este plano com o anterior, em simultâneo, usando a funcionalidade *Multiview*.

ADSyS - Sistema de Apoio à Decisão para Escalonamento

Questionário após a experimentação do protótipo ADSyS

	Caracterize a adequação do protótipo ADSyS à definição de um problema de escalonamento no que se refere	
	Ao dispositivo utilizado	
1	• Rato	péssima excelente |_|_|_|0|_|_|_|
2	• Teclas de atalho (teclado)	péssima excelente |_|_|_|0|_|_|_| NA |_|
	À estrutura geral adotada	
3	• Barra de menus	péssima excelente |_|_|_|0|_|_|_|
4	• Barra de ferramentas	péssima excelente |_|_|_|0|_|_|_|
5	• Janela de ferramentas	péssima excelente |_|_|_|0|_|_|_|
6	• Janela de atributos	péssima excelente |_|_|_|0|_|_|_|
7	• Barra de estado	péssima excelente |_|_|_|0|_|_|_|
8	• Área de trabalho	péssima excelente |_|_|_|0|_|_|_|
9	• Transição entre os diferentes editores	péssima excelente |_|_|_|0|_|_|_|

Sobre a definição de um problema de escalonamento com o protótipo ADSyS - Editor de problemas

10	• Inserção de tarefas	difícil |_|_|_|0|_|_|_|	fácil	
11	• Remoção de tarefas	difícil |_|_|_|0|_|_|_|	fácil	
12	• Seleção de tarefas	difícil |_|_|_|0|_|_|_|	fácil	
13	• Movimentação de tarefas	difícil |_|_|_|0|_|_|_|	fácil	
14	• Associação de uma gama operatória a uma tarefa	difícil |_|_|_|0|_|_|_|	fácil	
15	• Remoção de uma gama operatória associada a uma tarefa	difícil |_|_|_|0|_|_|_|	fácil	NA |_|
16	• Alteração dos atributos de uma tarefa	difícil |_|_|_|0|_|_|_|	fácil	NA |_|
17	• Alteração das datas de lançamento e de entrega, via manipulação com o rato	difícil |_|_|_|0|_|_|_|	fácil	NA |_|
18	• Definição do valor máximo do horizonte temporal	difícil |_|_|_|0|_|_|_|	fácil	
19	• Compreensão das mensagens que surgem como *tooltips*	difícil |_|_|_|0|_|_|_|	fácil	
20	• Identificação dos erros através da coloração dos contornos dos objetos	difícil |_|_|_|0|_|_|_|	fácil	
21	• Identificação dos erros através da coloração do interior dos objetos	difícil |_|_|_|0|_|_|_|	fácil	
22	• Compreensão do estado geral através da coloração do semáforo	difícil |_|_|_|0|_|_|_|	fácil	
	Relativamente à ferramenta de *Zoom*			
23	• Utilização	difícil |_|_|_|0|_|_|_|	fácil	NA |_|
24	• Utilidade	inútil |_|_|_|0|_|_|_|	útil	NA |_|
	Relativamente às ferramentas *Undo e Redo*			
25	• Utilização	difícil |_|_|_|0|_|_|_|	fácil	NA |_|
26	• Utilidade	inútil |_|_|_|0|_|_|_|	útil	NA |_|
	Relativamente às ferramentas *Cut, Copy e Paste*			
27	• Utilização	difícil |_|_|_|0|_|_|_|	fácil	NA |_|
28	• Utilidade	inútil |_|_|_|0|_|_|_|	útil	NA |_|
29	• Opinião global	péssima |_|_|_|0|_|_|_|	excelente	NA |_|

Sobre a construção de uma tarefa com o protótipo ADSyS - Editor de tarefas

30 • Inserção de operações — difícil |_|_|_|0|_|_|_| fácil

31 • Remoção de operações — difícil |_|_|_|0|_|_|_| fácil

32 • Seleção de operações — difícil |_|_|_|0|_|_|_| fácil

33 • Movimentação de operações — difícil |_|_|_|0|_|_|_| fácil

34 • Inserção de precedências entre operações — difícil |_|_|_|0|_|_|_| fácil

35 • Remoção de precedências — difícil |_|_|_|0|_|_|_| fácil

36 • Seleção de precedências — difícil |_|_|_|0|_|_|_| fácil

37 • Alteração dos atributos de uma operação — difícil |_|_|_|0|_|_|_| fácil

38 • Compreensão das mensagens que surgem como *tooltips* — difícil |_|_|_|0|_|_|_| fácil

39 • Identificação dos erros através da coloração dos contornos dos objetos — difícil |_|_|_|0|_|_|_| fácil

40 • Identificação dos erros através da coloração do interior dos objetos — difícil |_|_|_|0|_|_|_| fácil

41 • Compreensão do estado geral através da coloração do semáforo — difícil |_|_|_|0|_|_|_| fácil

Relativamente à ferramenta de inserção de precedências com operação de destino

42 • Utilização — difícil |_|_|_|0|_|_|_| fácil NA |_|

43 • Utilidade — inútil |_|_|_|0|_|_|_| útil NA |_|

Relativamente à ferramenta de *Auto-Layout*

44 • Utilização — difícil |_|_|_|0|_|_|_| fácil NA |_|

45 • Utilidade — inútil |_|_|_|0|_|_|_| útil NA |_|

Relativamente à ferramenta de *Zoom*

46 • Utilização — difícil |_|_|_|0|_|_|_| fácil NA |_|

47 • Utilidade — inútil |_|_|_|0|_|_|_| útil NA |_|

Relativamente às ferramentas *Undo* e *Redo*

48 • Utilização — difícil |_|_|_|0|_|_|_| fácil NA |_|

49 • Utilidade — inútil |_|_|_|0|_|_|_| útil NA |_|

	Relativamente às ferramentas *Cut*, *Copy* e *Paste*
50	• Utilização difícil fácil |_|_|_|0|_|_|_| NA |_|
51	• Utilidade inútil útil |_|_|_|0|_|_|_| NA |_|
52	• Opinião global péssima excelente |_|_|_|0|_|_|_| NA |_|
	Sobre a criação de várias máquinas destinadas à resolução de um problema de escalonamento com o protótipo ADSyS
53	• Inserção de máquinas difícil fácil |_|_|_|0|_|_|_|
54	• Edição de uma máquina difícil fácil |_|_|_|0|_|_|_|
55	• Apagar uma máquina difícil fácil
	Relativamente às ferramentas *Undo e Redo*
56	• Utilização difícil fácil |_|_|_|0|_|_|_| NA |_|
57	• Utilidade inútil útil |_|_|_|0|_|_|_| NA |_|
	Relativamente às ferramentas *Cut, Copy e Paste*
58	• Utilização difícil fácil |_|_|_|0|_|_|_| NA |_|
59	• Utilidade inútil útil |_|_|_|0|_|_|_| NA |_|
60	• Compreensão do estado geral através da coloração do semáforo difícil fácil |_|_|_|0|_|_|_|
61	• Opinião global difícil fácil |_|_|_|0|_|_|_|

Sobre a visualização do Plano de Escalonamento

62 • Movimentação de tarefas, via rato difícil fácil
|_|_|_|0|_|_|_|

63 • Movimentação de tarefas, via janela de atributos difícil fácil
|_|_|_|0|_|_|_|

Relativamente às ferramentas *Undo e Redo:*

64 • Utilização difícil fácil
|_|_|_|0|_|_|_| NA |_|

65 • Utilidade inútil útil
|_|_|_|0|_|_|_| NA |_|

Relativamente à diferenciação das tarefas através da coloração:

66 • Utilização difícil fácil
|_|_|_|0|_|_|_| NA |_|

67 • Utilidade inútil útil
|_|_|_|0|_|_|_| NA |_|

Relativamente à ferramenta *Save:*

68 • Utilização difícil fácil
|_|_|_|0|_|_|_| NA |_|

69 • Utilidade inútil útil
|_|_|_|0|_|_|_| NA |_|

Relativamente ao mecanismo de recuperação:

70 • Utilização difícil fácil
|_|_|_|0|_|_|_| NA |_|

71 • Utilidade inútil útil
|_|_|_|0|_|_|_| NA |_|

Relativamente ao modo dinâmico:

72 • Utilização difícil fácil
|_|_|_|0|_|_|_| NA |_|

• Utilidade inútil útil
|_|_|_|0|_|_|_| NA |_|

• Opinião global péssima excelente
|_|_|_|0|_|_|_| NA |_|

Relativo aos perfis de utilização e ao comportamento do sistema em relação a cada um desses perfis no protótipo ADSyS

73 • O modo de classificação dos utilizadores é apropriado? pouco muito |_|_|_|0|_|_|_| NA |_|

74 • Relativamente ao número de perfis de utilizadores deveriam existir: menos mais |_|_|_|0|_|_|_| NA |_|

75 • A transição entre os perfis principiante e intermédio deveria acontecer: mais cedo mais tarde |_|_|_|0|_|_|_| NA |_|

76 • A transição entre os perfis intermédio e avançado deveria acontecer: mais cedo mais tarde |_|_|_|0|_|_|_| NA |_|

As ajudas são claras e concisas no

77 • perfil principiante? pouco muito |_|_|_|0|_|_|_| NA |_|

78 • perfil intermédio? pouco muito |_|_|_|0|_|_|_| NA |_|

79 • perfil avançado? pouco muito |_|_|_|0|_|_|_| NA |_|

O nível de detalhe das ajudas no

80 • perfil principiante é adequado? pouco muito |_|_|_|0|_|_|_| NA |_|

81 • perfil intermédio é adequado? pouco muito |_|_|_|0|_|_|_| NA |_|

82 • perfil avançado é adequado? pouco muito |_|_|_|0|_|_|_| NA |_|

As ajudas são intrusivas no

83 • perfil principiante? pouco muito |_|_|_|0|_|_|_| NA |_|

84 • perfil intermédio? pouco muito |_|_|_|0|_|_|_| NA |_|

85 • perfil avançado? pouco muito |_|_|_|0|_|_|_| NA |_|

O número de ajudas no

86 • perfil principiante é: insuficiente suficiente |_|_|_|0|_|_|_| NA |_|

87 • perfil intermédio é: insuficiente suficiente |_|_|_|0|_|_|_| NA |_|

88 • perfil avançado é: insuficiente suficiente |_|_|_|0|_|_|_| NA |_|

89	• O modo de apresentação das ajudas é adequado?	pouco |_|_|_|0|_|_|_|	muito	
90	• As mensagens apresentam-se isentos de erros semânticos e gramaticais?	pouco |_|_|_|0|_|_|_|	muito	NA |_|
91	• A simbologia e as mensagens são consistentes e percetíveis?	pouco |_|_|_|0|_|_|_|	muito	NA |_|
92	• A linguagem utilizada é adequada ao público-alvo?	pouco |_|_|_|0|_|_|_|	muito	NA |_|
	Relativamente à utilidade das ajudas no				
93	• perfil principiante é:	inútil |_|_|_|0|_|_|_|	útil	NA |_|
94	• perfil intermédio é:	inútil |_|_|_|0|_|_|_|	útil	NA |_|
95	• perfil avançado é:	inútil |_|_|_|0|_|_|_|	útil	NA |_|
	Relativamente à janela de configuração das mensagens:				
96	• Utilização	difícil |_|_|_|0|_|_|_|	fácil	NA |_|
97	• Utilidade	inútil |_|_|_|0|_|_|_|	útil	NA |_|
98	• Este sistema é fácil de usar?	pouco |_|_|_|0|_|_|_|	muito	NA |_|
99	• É necessário uma demonstração do Sistema antes de usá-lo eficazmente?	pouco |_|_|_|0|_|_|_|	muito	NA |_|
100	• A maioria das pessoas iria perceber como funciona e como utilizar o sistema?	pouco |_|_|_|0|_|_|_|	muito	NA |_|
101	• Opinião global	péssima |_|_|_|0|_|_|_|	excelente	NA |_|
	Sobre a apresentação dos conteúdos do protótipo ADSyS				
102	• Os conteúdos são precisos e claros na forma como são apresentados	péssima |_|_|_|0|_|_|_|	excelente	NA |_|
103	• Os conteúdos apresentam-se Isentos de erros semânticos e gramaticais	péssima |_|_|_|0|_|_|_|	excelente	NA |_|
104	• A simbologia e as mensagens são consistentes e percetíveis	péssima |_|_|_|0|_|_|_|	excelente	NA |_|
105	• A linguagem utilizada é adequada ao público-alvo	péssima |_|_|_|0|_|_|_|	excelente	NA |_|
106	• Opinião global	péssima |_|_|_|0|_|_|_|	excelente	NA |_|

	Aspetos gerais sobre o protótipo ADSyS		
107	• O sistema é estável e corre sem falhas	péssima |_|_|_|0|_|_|_| excelente	NA |_|
108	• O utilizador dispõem de feedback sobre as suas ações	péssima |_|_|_|0|_|_|_| excelente	NA |_|
109	• Os menus encontram-se corretamente estruturados	péssima |_|_|_|0|_|_|_| excelente	NA |_|
110	• As opções nos menus são claras e concisas	pouco |_|_|_|0|_|_|_| muito	NA |_|
111	• As atividades repetitivas poderiam ser mais fáceis	pouco |_|_|_|0|_|_|_| muito	NA |_|
112	• As mensagens de erro são fáceis de perceber	pouco |_|_|_|0|_|_|_| muito	NA |_|
113	• As operações semelhantes são feitas da mesma forma em todo o sistema	pouco |_|_|_|0|_|_|_| muito	NA |_|
114	• Existem opções propensas a erros que podiam ser evitadas	poucas |_|_|_|0|_|_|_| muitas	NA |_|
115	• A navegação é consistente e intuitiva	pouco |_|_|_|0|_|_|_| muito	NA |_|
116	• O sistema informa sempre o que está a acontecer	pouco |_|_|_|0|_|_|_| muito	NA |_|
117	• A informação apresentada segue uma ordem lógica/natural	pouco |_|_|_|0|_|_|_| muito	NA |_|
118	• Existe informação apresentada desnecessária?	pouca |_|_|_|0|_|_|_| muita	NA |_|
119	• Opinião global	péssima |_|_|_|0|_|_|_| excelente	NA |_|

Na sua opinião o protótipo ADSyS é

120

difícil de usar |_|_|_|0|_|_|_| fácil de usar

frustrante |_|_|_|0|_|_|_| estimulante

rígido |_|_|_|0|_|_|_| flexível

121 Para a definição de um problema de escalonamento

nada familiar |_|_|_|0|_|_|_| muito familiar

inadequado |_|_|_|0|_|_|_| adequado

insuficiente |_|_|_|0|_|_|_| suficiente

114	**Críticas e sugestões ao sistema ADSyS**

Data __/__/____

Participante n.º __