An Agent-based Architecture to Support Adaptivity in Virtual Learning Environments Based on Learners' Learning Styles

Ph.D Thesis

Mohammad K. Al-Omari

A thesis submitted in fulfilment of the requirements

for the degree of Doctor of Philosophy

Centre for Computational Intelligence

Faculty of Technology

DE MONTFORT UNIVERSITY

2017

Declaration of Authorship

I, Mohammad K. Al-Omari, declare that this thesis titled, 'An Agent-based Architecture to Support Adaptivity in Virtual Learning Environments Based on Learners' Learning Styles' and the work presented in it are my own. I confirm that:

This thesis is written by me using LaTeX .

Signed:

_____

Date:

_____

Dedication

To my father

Prof. Khalid Al-Omari

Who encouraged me to do my Ph.D. Thank you for your endless support.

To my mother

Arwa Al-Omari

Without her prayers this work would have been impossible.

To my beloved wife

Mai Ramadan

Thank you for being with me during the tough times.

To my beloved children

Khalid, Laith and Hamza

Thanks for your interruptions, which have given me the insistence to complete this research.

Acknowledgements

Firstly, I would like to express my sincere gratitude to my first supervisor Dr. Jenny Carter for her continuous support throughout my Ph.D study, and for her encouragement and immense knowledge. Her guidance supported me during all stages of the research and writing of this thesis. I could not have imagined having a better supervisor for my Ph.D study.

I would also like to thank Prof. Francisco Chiclana for his insightful comments and encouragement. Thank you for the innovative ideas and brain storming sessions.

I thank my colleagues in the Centre of Computational Intelligence at De Montfort University for the exhilarating discussions and for all the fun we have had in the last three years. I would like to thank all of my friends especially Dwi Setiawan, Abdullah Al-Sokkar, Mohammad Al-Shirah, Amin Al-Ajlouni, Ayman Al-Omar, Haitham Al-Zaben and Issa Qabajeh.

Last but not the least, I would like to thank my family: my parents and my brothers and sisters for supporting me spiritually throughout the writing of this thesis and my life in general. I thank everyone who has supported me during my Ph.D journey.

Publications

1. Al-Omari, M., Carter, J., and Chiclana, F. (2016). A hybrid approach for supporting adaptivity in e-learning environments. *International Journal of Information and Learning Technology*, 33(5):333–348

2. Al-Omari, M., Carter, J., and Chiclana, F. (2015). A Proposed Framework to Support Adaptivity in Virtual Learning Environments. In *Proceedings of The European Conference on Technology in the Classroom 2015, Brighton, United Kingdom*, pages 257–264

3. Carter, J., Chiclana, F., and Al-Omari, M. (2015). E-Learning for Distance Students: A Case Study from a UK Masters Programme. In *Proceedings of The European Conference on Technology in the Classroom 2015, Brighton, United Kingdom*, pages 233–242

Abstract

Educational systems have been improving as a result of the revolution in technology. Learning Management Systems (LMSs) are becoming crucial and widely used in many educational institutions for the adoption of blended and distance learning. However, a "one-size-fits-all" approach is the basis of most of these systems, whereby differences and preferences such as the knowledge level and learning styles of learners are not taken into consideration in their design. Current e-learning systems are unable to provide learners with adaptive content that meets their learning styles preferences. This thesis aims to extend the capabilities of LMSs to support adaptivity based on learners' learning styles.

Therefore, a hybrid architecture design is proposed reflecting a novel approach in order to support dynamic real-time adaptivity in any LMS based on learners' learning styles. The architecture is designed based on a computational model using the technology of intelligent agents and the concept of the Event-Condition-Action (ECA) model. The proposed approach introduces a real-time dynamic adaptation process that follows specific adaptive features, namely, the type, number and order of the presented contents. These adaptive features are based on the recommendations of the Felder-Silverman Learning Styles Model. A system prototype of the approach is developed and integrated in an e-learning environment (i.e. Moodle). Finally, the proposed approach is evaluated using a case study in Moodle.

Contents

List of Figures

List of Tables

List of Abbreviations

AA	**A**daptive **A**gent
ACL	**A**gent **C**ommunication **L**anguage
AHMS	**A**daptive **H**yper **M**edia **S**ystem
AIA	**A**daptive **I**nterface **A**gent
ALMS	**A**daptive **L**earning **M**anagement **S**ystem
ALS	**A**daptive **L**earning **S**ystem
API	**A**pplication **P**rogram **I**nterface
CA	**C**ontrol **A**gent
CSA	**C**ourse **S**tructure **A**gent
ECA	**E**vent **C**ondition **A**ction
EH	**E**ducational **H**ypermedia
ERU	**E**vent **R**ecognition **U**nit
FSLSM	**F**elder-Silverman **L**earning **S**tyles **M**odel
GUI	**G**raphical **U**ser **I**nterface
IP	**I**nternet **P**rotocol
ITS	**I**ntelligent **T**utoring **S**ystem
JADE	**J**ava **A**gent **DE**velopment
LMS	**L**earning **M**anagement **S**ystem
LPA	**L**earner **P**rofile **A**gent
LSI	**L**earning **S**tyle **I**nventory
MAS	**M**ulti **A**gent **S**ystem
OU	**O**bservation **U**nit
PDA	**P**ersonal **D**igital **A**ssistant
REU	**R**ules **E**valuation **U**nit
TCP	**T**ransmission **C**ontrol **P**rotocol
VLE	**V**irtual **L**earning **E**nvironment

Chapter 1

Introduction

Objectives:

- Give an introduction and the motivation of this research
- Determine the research questions
- Present the research methodology
- Outline the structure of this thesis

1.1 Background

The revolution of technology and the internet has dramatically changed the teaching and learning process. E-learning has become one of the essential e-services that can be offered to communities. Many educational institutions have successfully adopted distance learning so that learners can learn anywhere at any time. Learning Management Systems (LMSs) such as Moodle and Blackboard are virtual learning platforms designed to facilitate and control the learning process. These systems are widely used by many educational institutions all over the world (Cosgrave et al., 2011; Walker et al., 2014). They provide learners with the required tools and features to help them to achieve their learning goals and to enhance the learning process; LMSs offer learning activities such as quizzes, exercises and discussion forums in addition to learning resources, which are the actual material of an online course. These activities and resources form a particular online course in the LMS.

Adaptivity in LMSs has emerged from Intelligent Tutoring Systems (ITSs) and Adaptive Hyper Media Systems (AHMSs), in which adaptive presentation and navigation are provided in these systems (Phobun and Vicheanpanya, 2010). Adaptivity can be based on different factors. For example, it can be based on learners' knowledge level, as well as learners' learning styles or disabilities. However, targeting instruction to the learner's abilities and needs can diminish course drop-out rates, improve learning outcomes and achieve learning goals (Pappas, 2015). Understanding learning styles and the role of learning styles in the teaching/learning process is a key component in effective teaching (Csapo and Hayen, 2006). Learners have different learning styles depending on their personalities and preferences. Consequently, the learning process can be enhanced when learners' preferences are considered in the design of LMSs. Most

of the current LMSs lack adaptivity support, which can, as a result, affect the learning process and prevent learners from achieving their learning goals.

Agent technology has been widely used in many fields especially in e-learning. An agent can be defined as a computer system that is capable of autonomous action in a particular environment to meet its design objectives (Wooldridge and Jennings, 1995). In the context of e-learning, the use of agents is a promising approach and has many advantages such as collaborative, adaptive and intelligent agents (Bokhari and Ahmad, 2013). It can be used to support adaptivity in e-learning systems. The Event-Condition-Action (ECA) model has been used in event-driven systems (Denecke, 2012). It is considered as a reactive model that responds in a timely manner to any events in a particular environment. LMSs as learning environments include many learners' events such as submitting a quiz, completing a questionnaire, etc. These events are stored in the database of the LMS. The ECA model can play a significant role in sensing the events in the LMS and reacting accordingly.

The focus of this research is to extend the capabilities of LMSs using the concept of agents and the ECA model in order to support adaptivity based on learners' learning styles.

1.2 Research Motivation

Adaptivity support based on learners' learning styles is one of the fundamental aspects of education and e-learning. It can enhance the learning process and help learners to achieve their learning goals. A "one-size-fits-all" approach is the basis of most of the current LMSs; all learners are provided with the same course content in the LMS and their individual learning styles and preferences

are not considered. As a result, learners may face difficulties in the learning process, which may affect their overall performance.

Several studies have proposed different frameworks and architectures, incorporating agent technology in order to support adaptivity in LMSs based on learners' learning styles. Some of these studies are still in the early stages of development whereas others are platform-dependent (i.e. for a specific LMS). Moreover, the process of changing adaptivity requirements may be time consuming and require restructuring of the LMS itself. To the best of the author's knowledge, no previous work has been done incorporating the concept of the ECA model and intelligent agents as a hybrid architecture to support adaptivity in any e-learning environment. The proposed architecture offers dynamic real-time adaptivity based on learners' learning styles in any LMS. The adaptation process is controlled using pre-defined pedagogical rules that can be defined and updated by teachers and course designers in order to achieve the learning outcomes. Hence, the motivation for this research on hybrid architecture is to support adaptivity in Virtual Learning Environments (VLEs) based on learners' learning styles.

1.3 Research Questions

The main questions that are investigated in this thesis can be summarised as follows:

- How can intelligent agents and the ECA model concept be utilised as a hybrid architecture in any e-learning environment?
- Does the proposed architecture support dynamic adaptivity in a timely manner based on learners' learning styles in the e-learning environment?

- What is the significance of the proposed architecture compared with related work?

1.4 Original Contributions

In this thesis, the main contribution is a hybrid architecture to support adaptivity based on learners' learning styles in e-learning environments. The architecture has been designed using intelligent agents and the ECA model. Moreover, a novel approach is proposed based on the hybrid architecture to provide dynamic adaptivity in a timely manner in the learning environment. Different course contents are presented to each learner based on his/her learning styles. The clarifications of the original contributions are summarised in the following points:

- A computational model has been designed to determine the main computational components of the proposed approach. It also describes the behaviour of each component.
- A hybrid architecture has been designed based on the computational model. It consists mainly of the multi-agent module, the ECA module and the e-learning environment. The ECA module is represented as database triggers in the database of the e-learning environment, which provides real-time events to the multi-agent module. The multi-agent module reacts to these events accordingly and provides learners with adaptive content based on their learning styles.
- A generic adaptive course structure has been designed. The structure consists of two main sections, namely the adaptive and non-adaptive sections. The adaptation process is based on the recommendations of the

Felder-Silverman Learning Styles Model (FSLSM) (Felder and Silverman, 1988). The process is controlled by proposed adaptive features, which can be updated by teachers and course designers.

- A system prototype has been developed and integrated in an e-learning environment (i.e. Moodle). The system prototype has been developed using Java Agent DEvelopment (JADE) as the development platform for the intelligent agents, and database triggers techniques, which are implemented in the database of the e-learning environment.
- A case study in Moodle with a set of different scenarios has been adopted to evaluate the proposed approach. The evaluation scenarios are used for the validation of the supporting dynamic adaptivity in a timely manner in the e-learning environment.

1.5 Research Methodology

It is vital to choose an appropriate research method to undertake any type of research. Researchers must determine precisely the suitable methodology from the early stages of the research . This provides a clear vision in regard to how to plan and carry out a particular research project. The constructive research method has been used to plan and present the research undertaken in this thesis. The constructive research method is commonly used for research undertaken in the field of Computer Science, Software Engineering and Information Systems (Crnkovic, 2010; Lukka, 2003). According to Crnkovic (2010), the constructive research approach implies building artefacts that can be practical, theoretical or both, in order to solve a domain specific problem. It involves the design and the development of solutions for particular problems. In constructive research, the newly developed constructs, models, algorithms

and techniques are referred to as contributions to knowledge (Crnkovic, 2010). In the field of Computer Science, many researchers have used the constructive research method in order to develop novel approaches that solve a particular domain problem. For example, Alghamdi (2013) proposed a novel policy-based runtime tracking approach to support learners in e-learning environments.

Based on the nature of this research, the author chose the constructive research method, as it fulfils the requirements of the research described in this thesis. It clearly identifies the main tasks in order to carry out this research including the development, implementation and evaluation of the proposed approach. Labro and Tuomela (2003) have identified a list of steps involved in the constructive research method. These steps are presented in the following points with an adjusted order to describe the work presented in this thesis:

1. Determine a specific domain problem with enough research potential.
2. Conduct an intensive literature review.
3. Construct a solution approach.
4. Show that the developed approach works.
5. Identify the research contribution of the approach.

The research conducted in this thesis proposes a novel hybrid approach to support adaptivity in e-learning environments based on learners' learning styles. By following the constructive research steps mentioned above, the proposed approach was utilised as shown in Table 1.1.

TABLE 1.1: The constructive research steps with respect to the thesis chapters

Constructive research step	**Related chapter**
Step 1: Determine a specific domain problem with enough research potential	**Chapter 1:** Introduction
Step 2: Conduct an intensive literature review	**Chapter 2:** E-learning systems and learning styles **Chapter 3:** Agent-based systems and the ECA model
Step 3: Construct a solution approach	**Chapter 4:** Architecture **Chapter 5:** System Prototype
Step 4: Show that the developed approach works	**Chapter 6:** Evaluation: A case study in Moodle
Step 5: Identify the research contribution of the approach	**Chapter 7:** Conclusion

1.6 Thesis Outline

The following points outline the next chapters in this thesis:

- **Chapter 2:** E-learning Systems and Learning Styles

  This chapter reviews the state of e-learning and e-learning systems. The limitations of current e-learning systems are identified and highlighted. The second part of this chapter investigates the learning styles models in the literature and their role in adaptation. Finally, an intensive review of learning systems incorporating learning styles is conducted in order to

understand the concept of adaptivity in learning systems and to identify the limitations of these systems.

- **Chapter 3:** Agent-based Systems and the ECA Model

  This chapter discusses agent technology in detail. It describes the taxonomy of agents and the environments where they can live. Moreover, the agents' development platforms are thoroughly investigated including the design methodologies used for building multi-agent systems. This chapter also provides a review of the most relevant systems incorporating agent technology to support adaptivity in e-learning environments. Finally, it introduces the ECA model and describes its application in e-learning environments.

- **Chapter 4:** Architecture

  In this chapter, a computational model is introduced. The model describes the main computational components of the proposed approach and their behaviour. Based on this model, the architecture of the proposed approach has been designed to support adaptivity in e-learning environments. It consists of three main components, namely, the LMS, the ECA module and the multi-agent module. Each component is discussed in detail. Finally, the adaptive features offered by the proposed approach are discussed based on the recommendations of the FSLSM.

- **Chapter 5:** System Prototype

  This chapter represents the actual implementation of the proposed approach. A system prototype is developed using the technologies introduced in Chapter 3. The components of the system prototype are discussed. This chapter also provides a mechanism for detecting the events

of LMS so that the multi-agent system can sense these events in a timely manner.

- **Chapter 6:** Evaluation: A Case Study in Moodle

  The system prototype is integrated and evaluated in Moodle, and this is presented in this chapter as a case study. A validation tool is developed in order to observe the results of each step in the adaptation process. A set of different live scenarios is used to validate the proposed approach. Finally, the significance of the proposed approach is discussed.

- **Chapter 7:** Conclusion and Future Work

  Chapter 7 is the last chapter of this thesis. It concludes the research and states future directions.

Chapter 2

E-learning Systems and Learning Styles

Objectives:

- To review the current state of e-learning systems
- To discuss adaptivity in e-learning systems
- To review the current state of learning styles models
- To investigate some adaptive learning systems based on learning styles

---

2.1 Introduction

This chapter is the first part of the literature review. It aims to introduce e-learning and e-learning systems, pointing out the drawbacks of the current LMSs as e-learning platforms. Learning styles models are introduced and discussed in detail, and the most common learning styles models in the literature are identified. Moreover, several learning systems incorporating learning styles are investigated, highlighting the limitations of these systems.

2.2 E-learning

The revolution of the internet and its services has dramatically changed the process of teaching and learning in higher education institutions all over the world; virtual environments are used as learning environments in order to simulate classroom and traditional learning. E-learning has become one of the most interesting e-services (e.g. e-commerce, e-government etc.) provided through the internet (Anghel and Salomie, 2003; Noaman et al., 2017). It offers new methods and facilities for both learners and teachers to develop and enrich their learning and teaching experience through the learning environment.

Although there are many definitions for e-learning in the literature, most of them focus on the same features that can be provided in e-learning environments. E-learning has been described as a new learning path that uses various technologies (Clark and Mayer, 2011; London, 2011; Tomic et al., 2011).The use of new technology has facilitated the learning process so that learners can learn anywhere at any time (Gros and García-Peñalvo, 2016). Furthermore, e-learning can be seen as the method of delivering knowledge and training using any electronic media (Li et al., 2009; Zimmermann, 2011; Hayakawa et al.,

2012). Electronic media is used for the delivery of content in order to facilitate the learning process. The philosophy of e-learning is to provide learners with tools and facilities to allow them to learn anywhere at any time; it entails great flexibility in terms of knowledge acquisition. According to Ellis et al. (2009), e-learning is the use of information and communication technologies in order to help learners achieve their learning goals.

The use of new technologies in e-learning can facilitate the learning process. However, it is important to improve the quality of learning by using these technologies. McGill et al. (2014) have stated that e-learning should improve the quality of learning and teaching in order to be successfully adopted.

The demand for teaching and learning via the internet is rising in many higher education institutions worldwide (Martínez-Caro et al., 2015). The use of e-learning platforms (i.e. Moodle and Blackboard) has become a strategy for achieving flexible online education. In order to meet today's increasingly personalised online-learning requirements, e-learning needs to provide more flexibility and dynamism (Graven et al., 2006; Kovalan and Balasubramanian, 2008; Dias et al., 2014). E-learning systems are used to deliver learning materials and to provide students and teachers with the required services and facilities in order to foster the learning process.

2.3 E-learning Systems

Recently, e-learning systems have become a paramount part of e-learning in many institutions such as universities, schools and training centres. Virtual Learning Environments (VLEs) as a type of e-learning systems are widely used in many higher educational institutions these days (Cosgrave et al., 2011; Walker et al., 2014).VLEs can be defined as platforms for delivering learning materials

to students via the internet. These systems are designed to provide tools and facilities (e.g. Discussion Forums, Online Assignments, etc.) to support learners and teachers and also to enhance the learning and teaching process. Moreover, VLEs are designed to be accessed both on and off-campus, which enables institutions to implement not only traditional face-to-face learning but also online learning for students who cannot attend on-campus classes due to time or geographic constraints. These systems can be implemented for the adoption of distance, blended and traditional learning. There are other terms that are used interchangeably in regard to this type of system, such as Course Management System (CMS) and Learning Management System (LMS). Moodle, Blackboard and Prometheus are examples of modern VLEs (Mallon, 2010). Nevertheless, there are different types of VLE that can be implemented in different educational institutions.

2.3.1 Types of Virtual Learning Environments

Although there are different types of VLEs, they perform the same function and can deliver the same learning content. VLE can fit into any one of the following three categories (Oxford University Press, 2017):

- Off-the-shelf (not free, licensed systems) such as Blackboard and WebCT
- Open source (often free to use) such as Moodle
- Bespoke (developed in-house to meet a specific institution's requirements).

Some of the above-mentioned e-learning platforms are introduced in Section 2.4. In order to maintain the interoperability in these systems, there are some universal standards associated with VLEs (i.e. they can be implemented in different types of VLE), namely, 'Sharable Content Object Reference Model' (SCORM) and 'Question and Test Interoperability' (QTI). The former is the

standard for content whereas the latter is the standard for assessments. SCORM is used for content in order to facilitate the design of courses so that the content can be transferred from one system to another without re-structuring it. On the other hand, QTI is used for assessments and building exams that can be easily exported and implemented in other systems.

VLEs have many tools and features that can be used by both learners and teachers to facilitate and support the learning and teaching process. Some of these tools and features are shown in Table 2.1.

TABLE 2.1: Tools and features of VLEs.

(O'Leary and Ramsden, 2002)

Communication Between Tutors and Students ✓ Email ✓ Discussion Forums ✓ Chat ✓ File Exchange ✓ VoIP (Video and Audio Services)	**Management and Tracking of Students** ✓ Registration ✓ Authentication ✓ Course Authorisation ✓ Student Tracking
Self-Assessment and Summative Assessment ✓ Multiple-choice Assessment ✓ Feedback ✓ Automated Testing and Marking	**Curriculum Design** ✓ Customised Look and Feel ✓ Instruction Standards Compliance ✓ Interoperability ✓ Course Templates

The tools and features provided in VLEs can vary from one platform to another depending on the platform provider. Moreover, the use of these tools and features can support learning and teaching (O'Leary and Ramsden, 2002). However, there are advantages and disadvantages of using VLEs.

2.3.2 Advantages and Disadvantages of Using VLEs

As mentioned earlier in this chapter, the use of VLEs is crucial for the adoption of distance and blended learning. However, there are both advantages and disadvantages of using these systems. According to O'Leary and Ramsden (2002), there is no intrinsic educational value in these systems as they depend upon the development of technology. Increasing the effectiveness and educational value of these systems depends on the way in which the online courses and activities are designed and presented. Some of the advantages and disadvantages of using VLEs are outlined in Table 2.2.

2.4 Examples of Current E-learning Systems

Learning Management Systems (LMSs) are widely used in many educational institutions such as universities, schools and training centres for the adoption of distance and blended learning. Moodle and Blackboard are two of the leading e-learning platforms (Pitigala Liyanage et al., 2013). In the UK, for example, the most commonly used LMSs are Blackboard, WebCT and Moodle (Oxford University Press, 2017).

TABLE 2.2: Advantages and Disadvantages of VLEs
(O'Leary and Ramsden, 2002; Cook, 2007)

Advantages	Disadvantages
✓ Easy online delivery of material	✓ Lack of face to face interaction
✓ Easy to use both for teachers and students	✓ Lack of student motivation
✓ Widens student access to learning materials and resources	✓ Lack of efficient teachers
✓ Discussion and support with students online	✓ Development of high quality learning material is time consuming and expensive
✓ Applying flexible style of teaching and learning	✓ Need of information technology skills
✓ Supports active and independent learning	✓ Need of training to use
✓ Offers flexible support for educators who do not need to be in a fixed time or place to support and communicate with students	✓ Lack of adaptivity support

2.4.1 Moodle

Modular Object-Oriented Dynamic Learning Environment (Moodle) is an open source learning management system written in PHP language and distributed under the GNU general public licence (Moodle.org, 2017). Moodle is a web-based platform that can be used for distance, blended and traditional learning. Moreover, it provides a powerful set of learner-centric tools and a collaborative environment in order to empower both the teaching and learning processes (Moodle.org, 2017). Moodle is probably the most popular and commonly used platform, with over 53,346 sites providing for over 70 million users across 222 countries (ibid). It can be used any time, anywhere, on any device that has an internet connection. A snapshot of Moodle is shown in Figure 2.1.

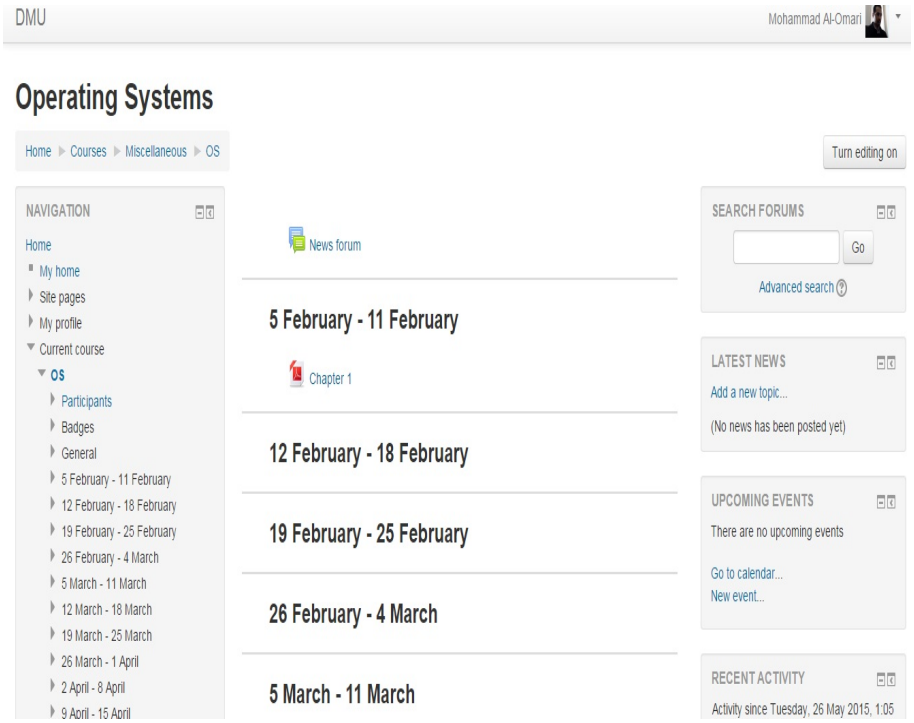


FIGURE 2.1: A Course Page in Moodle

2.4.2 Blackboard

Blackboard is a virtual learning environment developed by Blackboard Inc. It is a web-based and student-centred learning management system that provides learners and teachers with a variety of tools and features. Furthermore, Blackboard is a powerful learning platform implemented by many educational institutions worldwide. It has many features such as course management, grading and assessment, and collaboration and serves over 19,000 clients in 100 countries, including 1,900 international institutions (Blackboard.com, 2017). A snapshot of Blackboard is shown in Figure 2.2.

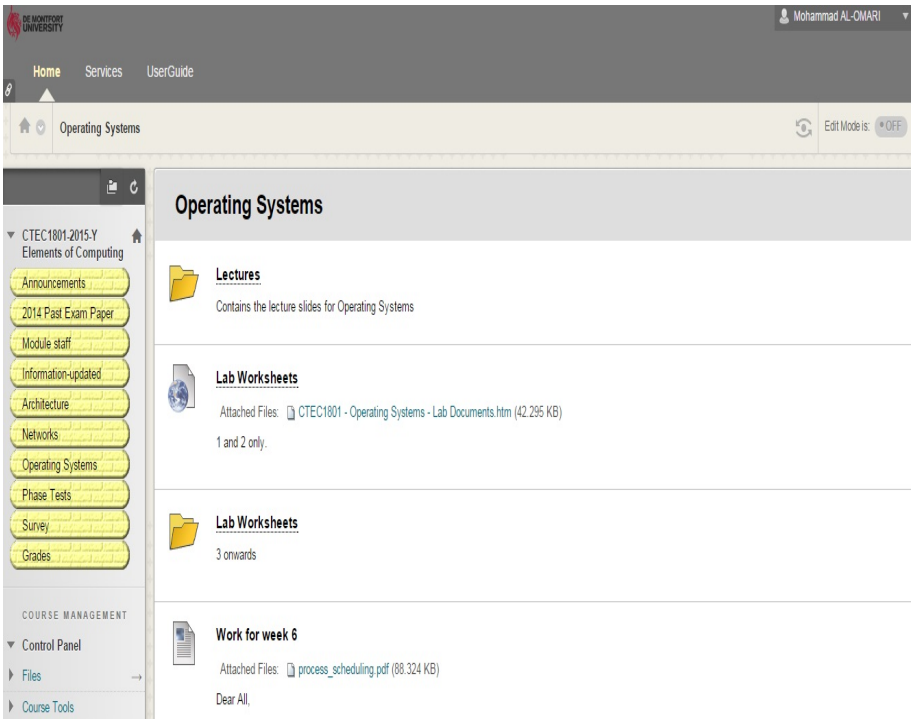


FIGURE 2.2: A Course Page in Blackboard

However, there is another LMS, namely WebCT; it is an online proprietary VLE that is widely used in many educational institutions. It is currently owned by Blackboard (Blackboard.com, 2017). These systems are some examples of current LMSs. However, Moodle and Blackboard are two of the leading LMSs that many universities use for the adoption of distance and blended learning.

2.4.3 Discussion

LMSs provide powerful tools to both teachers and learners in order to enhance the teaching and learning processes. Moodle and Blackboard, for example, provide a prodigious variety of features such as online assignments, discussion boards (forums), asynchronous and synchronous communication, quizzes, wikis etc. Moreover, these systems are web-based and designed to support

instructors and course designers in managing and designing online courses. Therefore, they have become very effective in technology enhanced learning and many educational institutions rely on these systems for the adoption of distance and blended learning.

LMSs are built based on pedagogical strategies that adhere to learning theories. Behaviourism, cognitivism, and constructivism, for example, are some of the concepts used for designing LMSs. The pedagogical strategies may vary from one system to another depending on the approach used by the developers. However, developing an LMS based only on a specific pedagogical strategy seems to be challenging, as few LMSs follow this track. For example, Moodle has been designed and developed considering social constructionist pedagogy, which encloses four concepts, namely, constructivism, constructionism, social constructivism, and connected and separate behaviour (Moodle.org, 2017). From a constructivist point of view, new knowledge is gained when learners interact with the system and use the previously possessed knowledge. Constructionism states that learning can be efficient when generating ideas for others to experience. Social constructivism expands constructivism into social settings using group discussions. Separate and connected behaviour deals with the motivation between learners during a group discussion. With separate behaviour, learners tend to defend their own ideas using logic and try to find weaknesses in their opponents' ideas. In connected behaviour discussions, learners accept subjectivity and try to accept each other's point of view. Nevertheless, the pedagogical strategies applied in LMSs follow the general point of view in teaching without taking into consideration the differences between learners, especially their learning styles. Thus, most of the current LMSs lack supporting adaptivity and learners' differences, such as their preferred learning styles and knowledge levels, have not been considered in the design of these systems.

2.4.4 Adaptivity in E-learning Systems

A "one-size-fits-all" approach is the basis of most of the current e-learning systems and differences and preferences such as the knowledge level and learning styles of learners have not been taken into consideration in their design. The idea of providing adaptive content in LMSs has emerged from Intelligent Tutoring Systems (ITSs) and Adaptive Hyper Media Systems (AHMSs). They provide both adaptive presentation and navigation. They are normally used in computer-based instruction (Phobun and Vicheanpanya, 2010). These systems are discussed in the following section.

2.5 Intelligent Systems for E-learning

In the literature in this field, there are two types of intelligent learning systems: Intelligent Tutoring System (ITS) and Adaptive Hypermedia System (AHS). These systems are able to act intelligently with learners and can simulate teachers to provide learners with the required support that can help them achieve their learning goals.

2.5.1 Intelligent Tutoring Systems

Intelligent tutoring systems (ITSs) are the enhanced version of traditional learning systems known as computer-aided instruction. They are developed and designed with the concept of Artificial Intelligence (AI) and its techniques. The design of ITSs is learner-centred, which means that more pedagogical knowledge is provided in the system (Ong and Ramachandran, 2000). ITSs are highly interactive learning environments with a one-to-one interaction in which learners

can practise their expertise. They assess learners' behaviour during the learning process and create models of their skills, expertise and knowledge (Phobun and Vicheanpanya, 2010).

In order to provide learners with course-related instructions, ITSs are composed of three types of knowledge, categorised into four software modules: the expert module, the instructional module, the learner module and the interface module, as shown in Figure 2.3. The expert module is a domain model that represents domain matter expertise. It supplies an ITS with the required knowledge in order to compare and evaluate learners' actions and behaviour. The instructional module is responsible for providing knowledge on how to teach and taking the required action to support students, such as providing feedback and guidance. Last but not the least, the learner module is used to evaluate each learner's performance during the interaction with the system in order to determine his/her perceptual abilities, skills and knowledge. The interface module is the intermediary between students and the system. It enables the students to interact with the system via its Graphical User Interface (GUI).

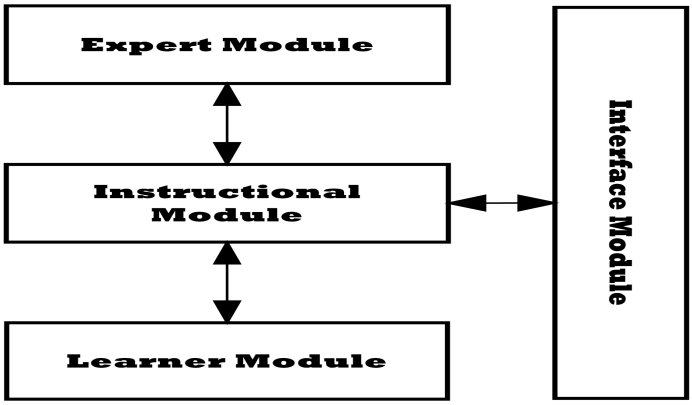


FIGURE 2.3: Components of an Intelligent Tutoring System (Phobun and Vicheanpanya, 2010)

ITSs are designed to provide practice-based learning and an interactive environment to enable learners to practise their skills and achieve their learning goals. These systems can assess each learner's activity and develop a model based on the skills, expertise and knowledge of the learner (Corbett et al., 1997; Ong and Ramachandran, 2003). Ong and Ramachandran (2003) argue that learners taught by ITS-based applications learn faster compared with classroom-based learners. For example, an experiment using the LISP tutor, an ITS that teaches computer programming skills to students, showed that students who used the system scored 43% higher on the final exam than students that received traditional instruction (Corbett and Anderson, 2001).

2.5.2 Adaptive Learning Systems

Adaptive learning can dynamically adjust the type of instruction based on each learner's abilities and knowledge, and can personalise the instruction in order to foster each learner's performance. Therefore, prevalent challenges such as a lack of resources, student motivation and the diversity of students' knowledge can be addressed during the learning process. Targeting the instruction to each learner's abilities and needs can diminish course drop-out rates, improve learning outcomes and facilitate the achievement of learning goals (Pappas, 2015). Musumba et al. (2013) argue that the current trends in education and training should focus on identifying tools and methods for delivering on-demand knowledge tailored to learners, taking into consideration their differences in terms of skills and knowledge level. Traditional Technology-Enhanced Learning (TEL) systems are designed with very few strategies in regard to the personalisation of the learning process. However, Adaptive Hypermedia Systems (AHSs) are developed in order to provide learners with a dynamic personalised learning environment. Brusilovsky defines Adaptive Hypermedia (AH)

as: "all hypertext and hypermedia systems which reflect some features of the user in the user model and apply this model to adapt various visible aspects of the system to the user" (Brusilovsky, 1996).

It can be inferred from the definition above that an AHS must have three characteristics: first, it must be a hypermedia system; second, it must follow a user model; and third, it must provide a personalised hypermedia presentation to each user or group of users based on this model. The user modelling and adaptation process in AHSs is shown in Figure 2.4.

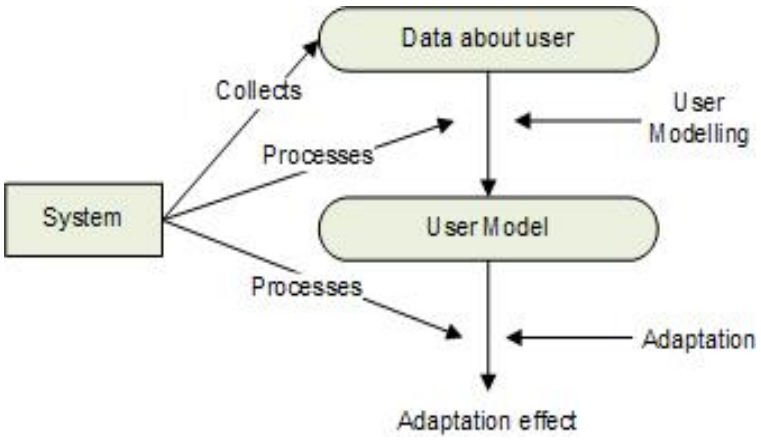


FIGURE 2.4: Classic loop 'user modelling – adaptation' in adaptive systems

(Brusilovsky, 1996)

As shown in the figure above, the system collects the required data about users either implicitly (by observing their behaviour with the system using data mining techniques for example) or explicitly (by asking them fill in a questionnaire, for instance). Then, the system processes the collected data and the data is represented and stored in a user model that can be updated when the user continues using the system. Finally, the adaptation effect is presented to users using an adaptive engine.

There are two different types of adaptive systems that should be taken into consideration: adaptive systems and adaptable systems. In adaptive systems, the adaptation process is done automatically without explicit intervention from users, whereas in adaptable systems, the adaptation process is done by providing users with tools to specify exactly how the system should act and how it should be altered, allowing for more control to be delegated to each user (Oppermann, 1994; Fumero, 2006).

Adaptive Hypermedia (AH) provides adaptivity, which contradicts the "one-size-fits-all" approach (Brusilovsky, 2000). In AHSs, the user model is the warehouse of the goals, knowledge and preferences of each individual user. Furthermore, it is used when users interact with the system; the adaptive content is presented to fulfil their needs. Basically, the AHSs are derived from the area of ITSs and are designed to merge the advantages of both ITSs and Educational Hypermedia (EH) (Brusilovsky, 2000). They are designed and developed based on the characteristics of both ITSs and EH.

Adaptation is the main issue in AH. Brusilovsky (1996, 2001) described two methods of providing adaptation in AH by identifying the taxonomy of Adaptive Hypermedia Technologies, namely, adaptive presentation and adaptive navigation. These methods are illustrated in Figure 2.5.

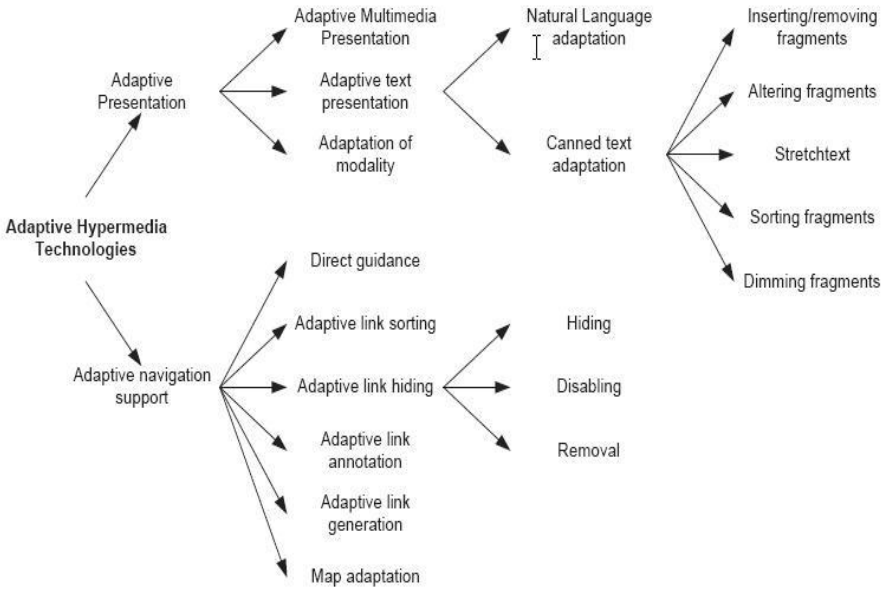


FIGURE 2.5: Taxonomy of Adaptive Hypermedia Technologies (Brusilovsky, 2001)

Adaptive presentation is the technique of customising content with the aim of fulfilling the needs of each user, such as stretch-text and conditional text; for example, a novice user is provided with general explanations and concepts whereas an expert user is provided with more detailed explanations and concepts. Adaptive navigation support deals with the way that links are displayed and presented to users to direct and guide them to accomplish their goals; it may restrict access, or sort and hide links. Moreover, it can be used for direct guidance during the learning process. Another type or category of adaptation is presented in Paramythis and Loidl-Reisinger (2004); this is the adaptive interaction type. This type is an adaptive interface in which adaptation occurs at the interface level of the system to support the user-system interaction without changing the content.

2.5.3 User Modelling

In order for adaptation to take place, data about users should be collected and stored in the database. This data is a crucial input for any adaptive hypermedia system. User modelling can be described as the process of generating a user model in which each user has a unique profile that reflects his/her characteristics and attributes. A user model can be defined as "a well-organized database which comprises information about the user and guides the system's inference engine" (Mylonakis and Cullough, 2003). Thus, it can be concluded that the user model is a model stored in a database that can be dynamically changed as the user interacts with the system to guide the adaptation process.

The adaptation process has been implemented in different systems. Some of these systems adapt to different aspects such as users' characteristics, including learning styles and preferences, while others adapt to usage and environmental data. As mentioned earlier, the user model is data about users that reflect their characteristics and goals. However, goals can be global, such as passing a course, or local such as completing a specific assignment. Furthermore, this data can reflect each user's knowledge level in a specific subject, as well as their preferred learning style, cognitive style and background (Santos et al., 2003).

Two of the most well-known user models are the overlay model and the stereotype model. According to Mette and Thomas (1994), the overlay model is based on the structural model of the subject. This technique of user modelling is able to measure each user's knowledge level. Moreover, this model can be described as a selected subset of the original domain model (Mylonakis and Cullough, 2003). On the other hand, the stereotype model deals with group formation in which each group has its own knowledge level, where users can

be assigned to a specific group depending on their knowledge and background. In other words, this model can be described as a cluster of users who share the same background and knowledge level. This model is dynamically updated so that it can decide whether the user should be transferred to another group based on specific criteria and conditions. There is also another model called the hybrid model, which combines the characteristics of both the overlay and stereotype models (Zakaria et al., 2002).

As mentioned earlier in this chapter, adaptivity in LMSs can be based on learners' knowledge levels and learning styles. Providing adaptive content based on learners' learning styles is one of the objectives of this research. Thus, the background of learning styles models is discussed in the following section.

2.6 Learning Styles

In the literature, there is no single definition of the learning concept. Learning can be defined as the process of active engagement with experience (MLA, 2017). It involves the development of new skills and knowledge as people interact with the environment. In other words, learning occurs when there is a change in our behaviour in which new skills and knowledge are acquired. However, dealing with previously acquired skills and knowledge is not a learning process; instead it is the act of reinforcing the previously acquired knowledge. Although the learning process involves acquiring new skills and knowledge, learners can learn differently according to their knowledge levels, personalities, skills and learning styles. The learning process should fit learners' needs in order to enhance the process and achieve its goals.

Learners have different learning styles depending on their personalities

and preferences. In the literature, no specific definition exists of the term learning style. Dunn and Dunn (1974) have defined learning style as "the manner in which at least 18 different elements from four basic stimuli affect a person's ability to absorb and retain". It is also defined as "a description of the attitudes and behaviours which determine an individual's preferred way of learning" (Honey and Mumford, 1992). Felder and Silverman (1988) have defined it as "characteristic strengths and preferences in the ways [students] take in and process information". Learning style may be seen as a psychological factor that may influence knowledge possession.

The term learning styles is sometimes used interchangeably with learning strategies; however, they are different. Learning strategies can be described as the methods used by learners to deal with specific actions during the learning process. As a result, learning styles can be extracted based on learning strategies (Pask, 1976). A learning strategy can be defined as "the way a student chooses to tackle a specific learning task in the light of its perceived demands" (Entwistle et al., 1979). On the other hand, learning style is defined as "a broader characterisation of a student's preferred way of tackling learning tasks generally" (Pask, 1976). Learning style might be closer in definition to cognitive style. Cognitive style is a technique used to organise and process information in a habitual and systematic manner (McLoughlin, 1999). It can be identified by testing learners' abilities and skills. McLoughlin (1999) has defined similar terms related to learning styles, as discussed in Table 2.3.

TABLE 2.3: Definitions of similar terms relating to learning styles (McLoughlin, 1999)

Term	Explanation
Learning strategy	Adopting a plan action in the acquisition of knowledge, skills or attitudes
Learning preference	Favouring one method of teaching over another
Learning style	Adopting a habitual and distinct mode of acquiring knowledge
Cognitive strategy	Adopting a plan of action in the process of organising and processing information
Cognitive style	A systematic and habitual mode of organising and processing information

Knowledge is perceived by learners in different ways. Jung (1923) outlined the major differences between individuals in terms of their perception, judgement and interaction. The study by Jung shows that people generally engage in one two mental functions: perceiving and judging (Myers and Briggs Foundation, 2017). Within each of these functions, he observed that people tend to perform that function using one of two methods, which are called preferences.

A learning style can be seen as "a preferential mode, through which a student likes to master learning, solve problems, think or simply react in a pedagogical situation" (University of Leicester, 2017). It reflects the way people learn, taking into consideration the differences among individuals in regard to learning in an appropriate learning environment in which the learner's need is fulfilled. Research had indicated that when the learning environment is consistent with students' learning preferences, students perform better (Ibid). In the literature, several learning styles models exist and these reflect the concept of learning styles. Some of these models are discussed in the following section.

2.6.1 Learning Styles Models

There are several learning styles models that have been used in research. Some of the most popular learning styles models are discussed in detail in the following sub-sections. These models are: Kolb's Learning Style Model, Honey and Mumford's Learning Style Model, the Dunn and Dunn Learning Style Model, the Myers-Briggs Type Indicator and the Felder-Silverman Model.

2.6.1.1 Kolb's Learning Style Model

According to Kolb's learning theory (1984), there are four different learning styles based on a four-stage circular learning process. Kolb's learning style model is based on John Dewey's experiential learning theory, Kurt Lewin's work focusing on the importance of being active in learning, and Jean Piaget's theory of cognitive development as the outcome of the transaction between people and the environment (e.g. education, business) (Kolb, 1984). Kolb's model provides both a method to grasp individual learning styles (preferences) and an explanation of the experiential learning cycle that applies to each learner. Within the cycle, Kolb proposes that each learner acts in a spiral movement of immediate experience in which observations and reflections on the experience are gained. Then, these reflections are linked to previously gained knowledge and converted into abstract concepts and theories that can be used as the backbone for evaluating applications of concepts in new situations. As a result, the concrete experience that closes the learning cycle is formed. Kolb's learning cycle is outlined in Figure 2.6.

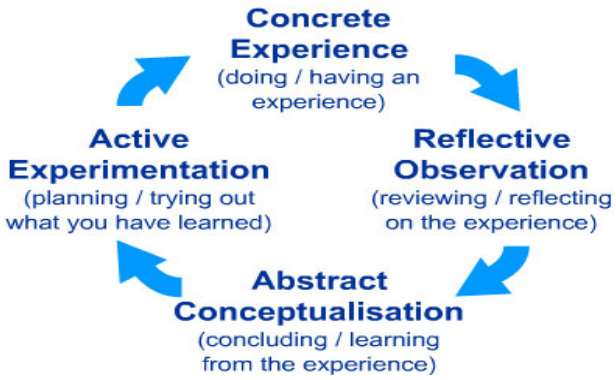


FIGURE 2.6: Kolb's Learning Cycle

(McLeod, 2010)

In order to identify the learning styles of this model, the Learning Style Inventory (LSI) has been created (Kolb, 1976). This inventory has a forced-choice ranking approach to determine learners' preferred styles in Kolb's cycle (Kolb and Kolb, 2005).

Learners are classified in Kolb's Learning Style model as active (learning through concrete experience), reflective (learning through reflective observation), experimental (learning through experimentation) and, theorising (learning through abstract conceptualisation).

2.6.1.2 Myers-Briggs Type Indicator (MBTI)

The personality factor created by Myers-Briggs (1962) reflects the four psychological functions, based on Jung's theory (Jung, 1923), by which people experience the world: sensation, intuition, feeling and thinking (Myers and Briggs Foundation, 2017). The MBTI is a personality test that is used to measure psychological preferences in regard to how people prefer to use their perception and judgment. It has four dichotomies that describe how people perceive and make decisions: Extraversion / Introversion, Sensing / Intuition, Thinking /

Feeling and Judging / Perceiving. These dichotomies are discussed in more detail in Table 2.4.

2.6.1.3 Honey and Mumford's Learning Style Model

Honey and Mumford (1992) adapted Kolb's model in order to create their own model of learning styles. The model has four stages that are similar to the learning cycle model: have an experience, review the experience, conclude from the experience and plan the next step. Honey and Mumford identified four different learning styles: Activist, Reflector, Theorist and Pragmatist. These learning styles are outlined in more detail in Table 2.5.

TABLE 2.4: The Myers-Briggs Type Indicator

(Myers, 1998)

Focusing attention	Extraversion(E)	Introversion(I)
	People who prefer extraversion tend to focus their attention on the outer world of people.	People who prefer introversion tend to focus their attention on the inner world of ideas and impression.
Taking in information	Sensing(S)	Intuition(I)
	People who prefer Sensing tend to take in information through the five senses and focus on present.	People who prefer Intuition tend to take in information from the big picture and focus on future.
Making decision	Thinking(T)	Feeling(F)
	People who prefer Thinking tend to make decisions based primarily on logic and on objective analysis of cause and effect.	People who prefer Feeling tend to make decisions based primarily on values and on subjective evaluation of person-centered concerns.
Dealing with outer world	Judging(J)	Perceiving(P)
	People who prefer Judging tend to follow a planned and organized approach to life.	People who prefer Perceiving tend to follow a flexible and spontaneous approach to life.

TABLE 2.5: Honey and Mumford Learning Styles

(Honey and Mumford, 1992)

Style	Attribute	Activities
Activist	Activists learn by doing and have an open-minded approach to learning, involving themselves fully and without bias in new experiences.	Problem solving Brainstorming Group discussions Puzzles
Reflector	Reflectors learn by observing and thinking. They prefer to view experiences and tasks from different perspective to work towards an appropriate conclusion.	Observing activities Coaching Interviews Questionnaires
Theorist	Theorists like to understand the theory behind the actions. They like discussions on theories and concepts. They use models and facts to engage in the learning process.	Background information Applying theories Models Statistics
Pragmatist	Pragmatists use new information and ideas to solve real-life problems. They need to be able to see how to put the learning into practice.	Problem solving Case studies Discussion

2.6.1.4 The Dunn and Dunn Learning Style Model

The Dunn and Dunn learning style model was originally proposed by Dunn and Dunn (1974). This model was refined and updated over a period of 33 years (Dunn and Griggs, 2007). It describes learning style as the way in which learners perceive, process and retain new information. The Dunn and Dunn learning style model is based on five variables that consist of different factors:

1. The environmental variable: This variable consists of light, temperature, sound, and furniture design.
2. The emotional variable: This variable includes motivations, conformity/ responsibility, task persistence, and need for structure.
3. The sociological variable: This variable includes sociological factors that deal with preference for learning alone, in a group, with an authority or with varied approaches.
4. The physiological variable: This variable includes the perceptual elements (auditory, visual, tactual and kinesthetic), consumption of food, and time.
5. The psychological variable: This variable was proposed later in the model. It includes analytical and global preferences, as well as impulsive and reflective preferences.

In order to detect learning style preferences according to this model, two inventories have been developed: the Learning Style Inventory and the Building Excellence Inventory. The former was developed for children (Dunn, R., Dunn, K., Price, 1996) and the latter was developed for adults (Rundle and Dunn, 2000).

2.6.1.5 The Felder-Silverman Leaning Style Model

The Felder-Silverman Learning Style Model (FSLSM) was developed by Felder and Silverman (1988) for two reasons. The first was to detect the most important learning style differences between engineering students, and the second was to offer a good foundation for teachers to design an approach to teaching in which the learning needs of learners would be fulfilled (Felder and Spurlin, 2005). In this model, learners' learning preferences are classified in four learning style dimensions: sensing / intuitive, visual / verbal, active / reflective or sequential / global (Ibid). According to Felder and Spurlin (2005), each of the dimensions has parallels in other models such as the models by Myers-Briggs (1962), Kolb (1984), and Honey and Mumford (1982). The combinations, however, are unique to Felder's work. These dimensions are explained in Figure 2.7.

The active / reflective dimension is similar to Kolb's perspective dimension. Active learners are fully motivated to work within groups. Moreover, they work actively with the learning material, trying to apply, discuss and explain it to others. On the contrary, reflective learners would rather think about the learning content and prefer to work individually.

The sensing / intuitive dimension has emerged from MBTI (Myers-Briggs, 1962). Sensing learners tend to learn facts and specific learning material by focusing on the details. Furthermore, they tend to be more realistic than intuitive learners and like to apply what they have learned to the real world. On the other hand, intuitive learners are more creative than sensing learners. They prefer to explore things and build relationships.

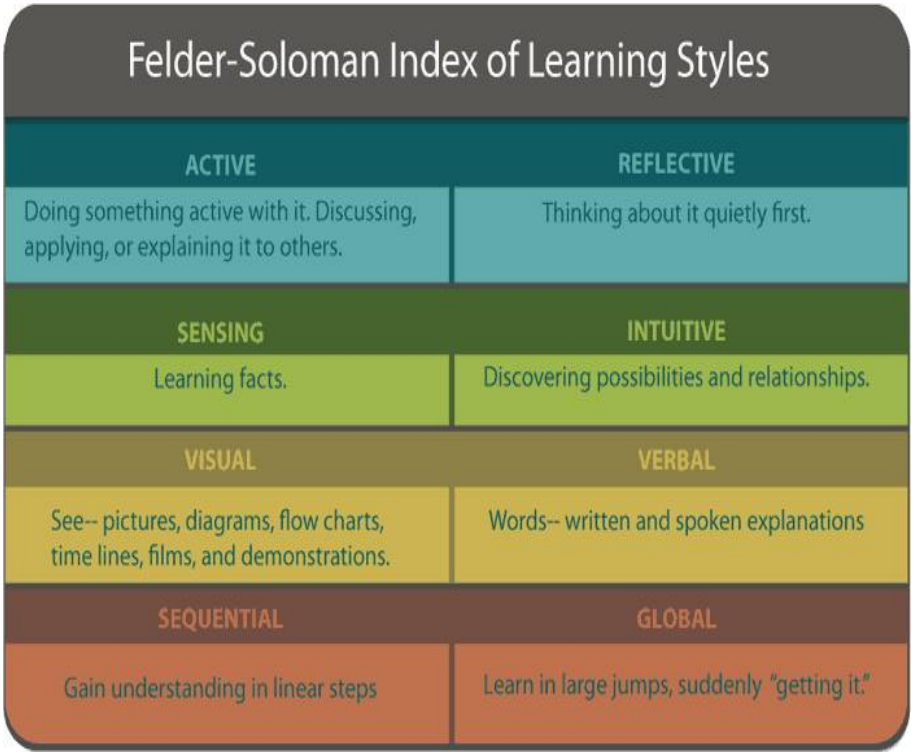


FIGURE 2.7: The four dimensions of Felder-Silverman Learning Styles Model

(The University of Texas at Austin, 2017)

The visual / verbal dimension deals with the presentation of content in order to help learners understand and remember it. Visual learners prefer to see pictures, diagrams, flowcharts and so on, whereas verbal learners favour textual (written or spoken) presentations.

The last, sequential/global dimension deals with the way in which learners understand the content. Sequential learners tend to learn in a serial manner in order to understand the material or concepts. In contrast, global learners prefer to look at the general picture of the concepts and tend to ignore the details and connections between these concepts.

To identify learning styles based on the FSLSM, the Felder-Soloman Index of Learning Styles (ILS) is utilised to identify each learner's preferred dimensions in the learning style model (Soloman and Felder, 2005). Learners are asked to respond to 11 forced-choice questions for each of the four dimensions (a 44-item questionnaire). Each question has two possible responses "a" or "b" and each response corresponds to one of the categories related to the dimension. The "b" responses are subtracted from the "a" responses to obtain a score; this will be an odd number between -11 to +11 (Felder and Spurlin, 2005).

2.6.1.6 Learning Styles Models Comparison

Different learning styles models have been used by many researchers. However, there are some dominant and popular learning styles in the literature. A study by Al-Azawei and Badii (2014) shows that the FSLSM is one of the most dominant and popular learning styles models in the literature. This is due to many reasons. One of these is the validity and reliability of its ILS (Riding, 1991; Kolb and Kolb, 2005; Viola et al., 2006; Litzinger et al., 2007). The model is consistent with learners' preferences (Villaverde et al., 2006). It is also suitable as a model for technology enhanced learning (Cha et al., 2006; Graf, 2007; Huang et al., 2012; Yang et al., 2013).

Although learning styles models can be used to capture learners' preferred learning styles, they have been criticised by theorists. Some theorists argue that learning styles are stable and not changeable over time. Others argue that learning styles are relatively flexible (i.e. they change over time). For example, Pask (1976) considers the serial and holistic learning style (which is related to the sequential and random style by Gregorc (1982)) as flexible, while Gregorc claims that learning styles remain unchangeable. The use of questionnaires as instruments to measure learning styles has also been criticised. However, it can

be concluded from the literature that a questionnaire that conforms with the Felder-Soloman Index of Learning Styles can be seen as a reliable questionnaire that can capture learners' learning styles.

Therefore, the FSLSM is used in our proposed approach in order to capture learners' learning styles. A questionnaire was developed and designed based on the Felder-Soloman Index of Learning Styles, as discussed later in Chapter 4. In the following section, some adaptive learning systems that incorporate learning styles are discussed.

2.7 Review of Adaptive Learning Systems Incorporating Learning Styles

There are many Adaptive Learning Systems (ALSs) in the literature in which different techniques for adaptation are implemented and used. The following systems are some examples of ALSs that incorporate learning styles.

2.7.1 Arthur

Arthur is a web-based learning system that presents adaptive content to learners based on learning styles. Course content is created and designed by a group of instructors from the same discipline. The content is based on several learning styles. In this system, the Case-Based Reasoning (CBR) technique is used to capture each learner's learning style (Gilbert and Han, 1999). When the learner logs into the system, the first concept in the course is provided randomly. Then, the learner is evaluated by accessing a quiz. Based on his/her results, the system can decide whether there is a match between the learning style used and

the learner. Thus, the results obtained can be used for future classification of learners (the concept of CBR). Arthur has been developed using Java Applets, Intelligent FAQ, Knowledge Base and SQL database. The system summary is shown in Figure 2.8.

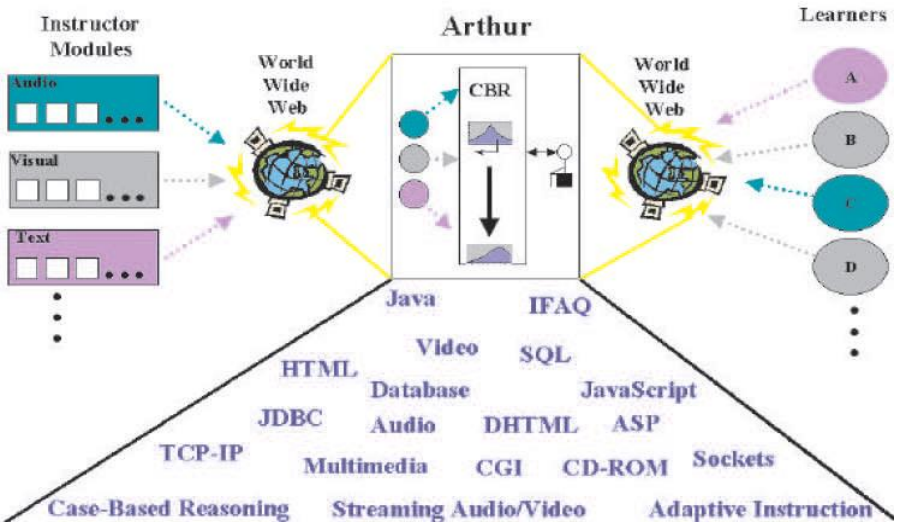


FIGURE 2.8: System Summary of Arthur

(Gilbert and Han, 1999)

2.7.2 iWeaver

iWeaver is an interactive web-based adaptive learning system in which adaptive features are provided (Wolf, 2003).The architecture of this system has been designed using the learning style model of Dunn and Dunn (Dunn and Dunn, 1974; Dunn and Griggs, 2007). It supports both adaptive navigation (link hiding and ordering) and adaptive presentation. This system has been developed using Java Server Pages (JSP), Java and MySQL as the Database Management System (DBMS). Moreover, it is used for teaching Java as a programming language. A combination method is used in order to find the best match between the media content and the Dunn and Dunn learning style to be presented to

learners. There are four types of learners based on the Dunn and Dunn learning style model: visual text, visual pictures, tactile (kinaesthetic) and auditory learners. Visual text learners are provided with rich text format content such as text documents. Visual picture learners are provided with content that contains diagrams and illustrations. Tactile kinaesthetic learners are provided with interactive content, whereas audio files and bulleted main points are provided for auditory learners. Furthermore, learning tools such as note-taking tools and extra examples tools are provided for each learner. When learners log into the system for the first time, they are asked to complete the "Building Excellence Inventory" (Rundle and Dunn, 2000) and assessed according to the Dunn and Dunn learning styles model. Then, based on their result, a preliminary student model is created. During the learner's interaction with the system, the student model is updated, refined and compared with the content model. The system architecture of iWeaver is shown in Figure 2.9.

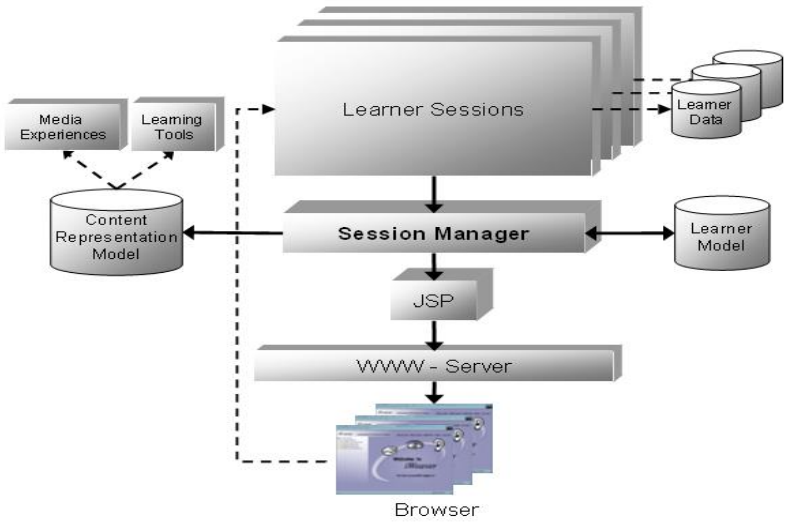


FIGURE 2.9: System architecture of iWeaver

(Wolf, 2003)

2.7.3 INSPIRE

INSPIRE stands for Intelligent System for Personalized Instruction in a Remote Environment. It adapts the lessons according to each learner's knowledge level, progress and learning style (Papanikolaou et al., 2003). Students are able to change their student model so this system can be considered as an adaptive (it provides adaptive content) and adaptable system (learners can explicitly update their preferences). It is based on Honey and Mumford's learning style model (Honey and Mumford, 1992). Two instructional theories are used in INSPIRE, namely, the Component Display Theory (Merrill, 1983) and the Elaboration Theory (Reigeluth and Stein, 1983).

This system provides adaptive navigation, curriculum sequencing and adaptive presentation techniques. The adaptive navigation and curriculum sequencing techniques are formed using learners' cognitive abilities and knowledge levels whereas the adaptive presentation is designed using their learning styles. The system can track the actions and behaviours of each learner. However, the learning styles are not captured using the data from the tracking process. Instead, the learners are provided with a survey developed by Honey and Mumford (1992) in which the initial student model is built and initiated. Moreover, this model can be explicitly updated by the learners. INSPIRE has been developed using Active Server Pages (ASP), SQL Server, ActiveX Data Objects (ADO).

2.7.4 CS383

CS383 is the earliest adaptive system. It was designed using the FSLSM (Carver et al., 1999). Two approaches are implemented in this system, namely, an adaptive testing system based on guidance and adaptive navigational support

through the adaptive sorting of content. Learners complete an online survey in which a student model is created in order to assess their learning style based on the FSLSM. Based on the results, the course material is presented and sorted according to its support for that model. Moreover, this system provides a vast collection of media objects such as digital clips, audio files, note taking guides, a digital library, lesson objectives and slide shows; each learner can sort these objects to fit their individual learning style. In addition, the learner can modify his/her learning style directly from his/her profile. This system has been developed using the Common Gateway Interface (CGI).

2.7.5 InterBook

This system is an authoring tool for the delivery of adaptive e-books on the World Wide Web (WWW). It affords an individual model for each user's knowledge and implements this model to afford adaptive navigation support, adaptive help and adaptive guidance. The InterBook approach is based on electronic textbooks, which can be presented in any hierarchically structured hypermedia material. According to Brusilovsky et al. (1998), the InterBook approach deals with two types of knowledge, namely, knowledge about the domain, which is called the domain model, and knowledge about the user, which is called the user model. The domain model is mainly used for structuring the content of the adaptive electronic textbooks. It contains the domain concepts, which are the basic elements of knowledge for the given domain. Furthermore, InterBook contains a glossary and indexed electronic textbooks. The knowledge about the learner is reflected in the grades of each concept, which might reflect the following different levels: Expert-Knowledge, Intermediate-Knowledge, Beginner-Knowledge, and No-Knowledge.

2.7.6 IDEAL

IDEAL stands for Intelligent Distributed Environment for Active Learning (Shang et al., 2001). It is an adaptive system developed using agents to provide and support active learning. This system has been designed to provide both navigation and content adaptation features. The content is adapted to students depending on their learning styles, knowledge, language and accessibility. The architecture of IDEAL provides many adaptation features in terms of presentation (icons and metaphors) and navigation (ordering links and content). However, these adaptation features should comply with the learning style model being used. The learning style capturing process is done using a questionnaire when the students register for a specific course. The knowledge level for each learner is frequently updated based on their progress and performance. The framework of IDEAL is shown in Figure 2.10.

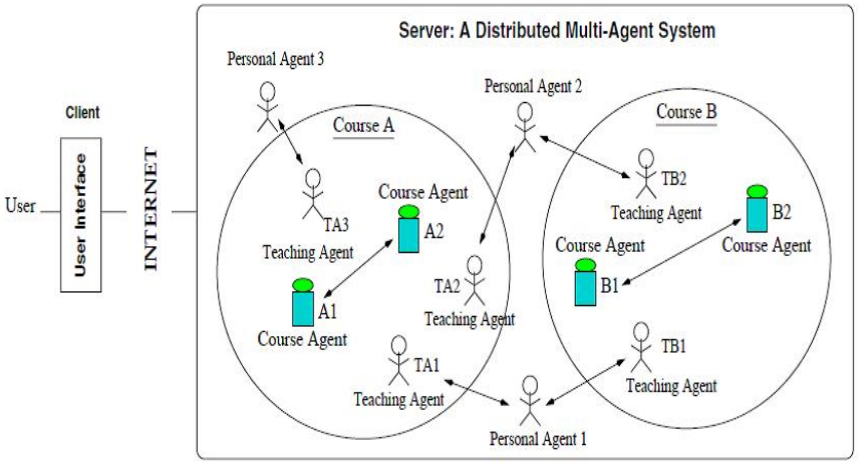


FIGURE 2.10: The framework of IDEAL, a Web-based interactive learning environment

(Shang et al., 2001)

2.7.7 AHA!

AHA stands for Adaptive Hypermedia for All; it is a free adaptive system written entirely in Java using Servlets, and developed at the Eindhoven University of Technology (AHA Project, 2016). An authoring tool is provided in the system that helps authors to implement a suitable learning style model (Bra and Calvi, 1998; Stash et al., 2006). Moreover, AHA provides both adaptive presentation and adaptive navigation to learners. Adaptive presentation deals with the content tier where the conditional inclusion of fragments is used in order to check whether or not the rules or the requirements are fulfilled. These adaptation rules control and define how the user model is refined.

In contrast, adaptive navigation is the adaptation at link level where link hiding and annotation are used to guide students to meet their learning goals. Learners can manually choose their learning style preferences from the predefined learning styles, which can also be revised and updated using learning meta-strategies created by the authors. Therefore, a learner's learning style can be inferred while the learner interacts with the system. The adaptation process in AHA is outlined in Figure 2.11.

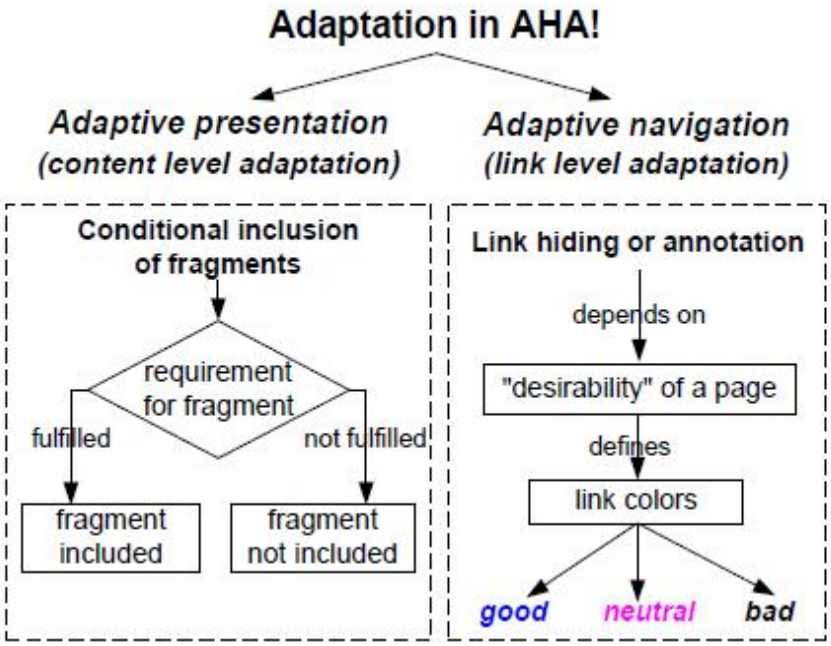


FIGURE 2.11: Adaptive techniques used in AHA!

(Bra et al., 2003)

2.7.8 Discussion

In this section, some of the adaptive hypermedia learning systems that incorporate learning styles have been introduced and discussed. These systems have been built and designed focusing on a learner-centred approach in which adaptive courses are presented to fit learners' needs. Although technologically all of the systems above are probably obsolete, they support adaptivity, which most of the current learning management systems are lacking. However, when these systems were applied in real teaching environments, some limitations emerged (Brusilovsky, 2004). The problem with these systems is their structure;

they were not designed to support teachers and administrators (ibid). Brusilovsky (2004) points out two issues concerning these systems: the lack of integration and the lack of interoperability. They offer only limited features in terms of web-enhanced education. Moreover, it is difficult for teachers to re-use the adaptive content and to apply it in a different system. Consequently, these platforms are rarely implemented in educational institutions these days.

Nevertheless, LMSs such as Moodle and Blackboard are widely used and implemented in many educational institutions worldwide. They provide powerful tools to support learners, teachers and administrators. Yet, most of the current LMSs lack adaptivity. Learners' differences, such as their preferred learning styles and knowledge levels, were not considered when these systems were designed. The focus of this thesis is to extend the capabilities of LMSs to support adaptivity based on learners' learning styles.

2.8 Summary

In this chapter, the current state of e-learning systems has been discussed and reviewed. First, we have defined e-learning and e-learning systems outlining the current challenges and limitations of the current VLEs. Supporting adaptivity is crucial in these systems so that learners' differences are taken into consideration. In the second part of our literature review, we have discussed some of the current learning styles models in detail, pointing out one of the most common and popular learning style models. Moreover, an intensive review has been carried out of some learning systems that incorporate learning styles in order to understand the concept of adaptation in e-learning environments. In addition, the limitations of these systems have been identified. The next chapter (Chapter 3), which is the second part of the literature review, introduces the technologies used in the proposed approach.

Chapter 3

Agent-Based Systems and Event-Condition-Action (ECA) Model

Objectives:

- To introduce agent technology and agent taxonomy
- To review the current agent platforms
- To review agent-related work in the context of adaptive e-learning systems
- To investigate the concept of the ECA model and to outline its significance in e-learning environments

---

3.1 Agent Based Systems

Agent-based systems are widely employed in many fields such as software engineering, artificial intelligence, distributed systems, social sciences and e-learning. Furthermore, they are applied in many systems – ranging from relatively simple systems such as personalised assistants to complex systems such as air traffic control (Wooldridge, 2009). Agent technology is considered a paramount approach in creating industrial collaborative distributed systems (Jennings et al., 1995). In the literature, there is no specific definition of agents. However, all of the definitions available concur that agents are autonomous; this is how they differ from objects. There is a broad agreement that autonomy is a central, distinctive property of agents. Wooldridge and Jennings (1995) define an agent as "a computer system situated in some environment, that is capable of autonomous action in this environment in order to meet its design objectives". Autonomy means that agents should be able to act in an environment without an explicit intervention from users and should have control over their actions. Although agents can be seen as objects (i.e. having properties, calling methods etc.), they are distinguished from objects in terms of their behaviour; agents are autonomous and have control over their actions. In contrast, objects use calling methods to accomplish specific tasks and they have no control over the execution of those methods (Jennings and Wooldridge, 1998). For example, if an object (A) calls a method (X) on an object (B), then (B) has no influence over whether (X) is performed or not. In other words, object (B) is not autonomous since it has no control over its own behaviour. Objects exhibit control over their state but not over their behaviour (i.e. they have a limited sense of autonomy). The basic abstract view of an agent is depicted in Figure 3.1.

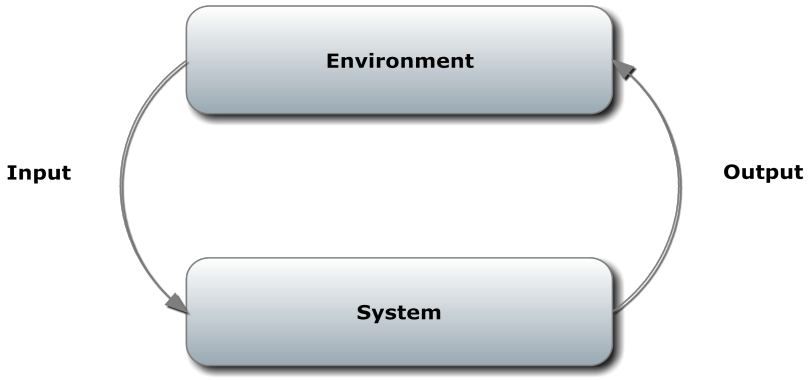


FIGURE 3.1: The basic abstract view of an agent

From this point of view, an agent can be considered as intelligent if it is capable of flexible autonomous action whereby its design objectives are met. Wooldridge (2009) has identified flexibility in the architecture of an agent such that an intelligent agent should have the following capabilities:

- Reactivity: intelligent agents should respond in a timely fashion to changes they perceive in their environment.
- Proactivity: intelligent agents can take the initiative to meet their design objectives, and they exhibit goal-directed behaviour.
- Sociability: intelligent agents can interact with other agents to meet their design objectives.

3.2 Multi Agent Systems

A Multi Agent System (MAS) can be considered as a collection of agents working together (i.e. collaboration between two or more agents) in order to achieve a final objective. According to Jennings et al. (1998), an MAS is a collection of autonomous agents working together in the same environment to achieve the

overall design objective of the system and to overcome the limitations of individual agents. Each agent in an MAS has specific tasks to accomplish. However, the agents should communicate and interact with each other to achieve the overall design objective of the system. There are many types of agents. These types are discussed in the following section.

3.3 Agent Types

In the literature, there are several types of agents and each type has its own characteristics. However, determining suitable types of agents depends on the environment that hosts these agents in addition to the objectives of designing a system. According to Nwana (1996), there are seven types of agents:

3.3.1 Collaborative Agent

Collaborative agents emphasise autonomy and cooperation with each other to accomplish their tasks in time-constrained and open multi-agent environments (Nwana, 1996). The main general characteristics of these agents include autonomy, social ability, responsiveness and proactiveness (ibid). These agents negotiate to reach an agreement, as will be discussed in section 3.5.2. Moreover, they are used to solve large scale problems that could not be solved using an individual agent. These agents are utilised to perform decentralised management and control of consumer electronics. The Multimedia Information Interchange (MII) system is an example of this type of agents (Titmuss et al., 1997).

3.3.2 Interface Agent

Interface agents collaborate with users rather than collaborating with other agents as in collaborative agents. However, interface agents can request recommendations from other agents. Maes (1994), a key advocate of this type of agent, has pointed out that the key structure of this type of agent is the personal assistant who collaborates with users in order to accomplish specific tasks. According to Nwana (1996), there are three generic benefits of using interface agents. Firstly, they reduce the workload of users and developers. Secondly, they can adapt to users' preferences. Finally, they have reasoning in regard to how they may be shared among users. An example of this type of agent is a personalised recommendation system for music albums and artists, the Ringo/HOMR system (Shardanand and Maes, 1995).

3.3.3 Mobile Agent

Mobile agents can be defined as software processes that are capable of wandering in the World Wide Web, communicating with other hosts, collecting information and getting back to their original destination after their tasks are accomplished (Nwana, 1996). The first mobile agent application was Sony's Magic Link PDA or personal intelligent communicator (PIC); it is used to manage a user's email, fax and phone.

3.3.4 Information / Internet Agent

With the explosion of the WWW, information agents play a significant role in managing the huge amount of data that users may deal with. These agents are

designed for managing, manipulating or collating information from different sources of data (i.e. different locations) (Nwana, 1996).

They can act as mobile agents when needed in the web. Nwana (1996) has stated that these agents may be able to pass through the WWW using indexers (pointers) to gather information and report back to the original location. A prototypical example of this kind of agent is the Internet Softbot (Etzioni and Weld, 1994).

3.3.5 Reactive Agent

Reactive agents are autonomous agents that are capable of sensing the environment and reacting accordingly. They respond in a stimulus-response manner in the environment in which they are located, and they have no internal symbolic models of their environments (Nwana, 1996). Moreover, they have some advantages such as simplicity, economy and robustness against failure.

A reactive agent can be seen as a group of modules in which these modules can operate together to accomplish specific tasks in order to achieve a final goal. This type of agent can be used in 3-D graphics-based and reactive agent animations (Wavish, 1996).

3.3.6 Hybrid Agent

The architecture of this type of agent is based on a combination of two or more agent types (Nwana, 1996). Therefore, hybrid agents are designed in a way to merge the benefits of more than one agent type for applications that need such a combination. An example of a hybrid system is the Cooperative Intelligent Real-time Control Architecture (CIRCA) (Musliner et al., 1993). There is also

another type of agent, namely, the Heterogeneous Agent. Heterogeneous agent systems refer to the integration of at least two or more agents that belong to distinct agent classes (Nwana, 1996). They may have various hybrid agents within their architecture.

Although there are various types of agents, they must achieve the system's objectives and goals in a specific environment. There are different types of environments in which agents may be implemented. These types are discussed in the following section.

3.4 Types of Environments

According to Russell and Norvig (1995), the types of environments in which agents may find themselves are as follows:

3.4.1 Accessible versus Inaccessible

An agent is able to sense complete and up-to-date information in accessible environments. As a result, within these environments, the internal state of an agent remains unchanged when the agent tracks the world (Russell and Norvig, 1995). On the other hand, inaccessible environments such as the Internet are not accessible in this sense (Wooldridge, 2009). However, the more inaccessible an environment is, the more complicated it is to build agents to act in it.

3.4.2 Deterministic versus Non-deterministic

An environment is called deterministic when the following state of the environment is known without uncertainty; when an agent performs an action, the

effect is determined and guaranteed. Russell and Norvig (1995) have stated that "If the environment is inaccessible, then it may appear to be nondeterministic". This can be seen as true in complex systems, especially when the process of tracking is difficult. However, tracking ability in an environment may be seen as a key aspect that may reflect the type of environment.

3.4.3 Static versus Dynamic

A static environment can be defined as an environment that remains unchanged unless an agent performs some actions. Therefore, time and changes are not taken into consideration when agents perform their actions in this type of environment. A dynamic environment, such as the Internet, in contrast, has other processes that are outside of agents' control (Wooldridge, 2009).

3.4.4 Discrete versus Continuous

A discrete environment is an environment that has a fixed and limited number of actions that may occur whereas a continuous environment has an infinite number of operations and actions. Environments are called complex when they are inaccessible, non-deterministic, dynamic and continuous (Wooldridge, 2009). As a result, implementing agents in these environments can be extremely complicated.

3.5 Agent Communications

Agents communicate with other agents by passing messages between them in order to achieve their design objectives. The communication process involves

the sending of messages (actions) and the receiving of messages (perceptions) using specific standards (Weiss, 2013). There are two kinds of communication, negotiation or co-operation. The former occurs between competitive agents or self-interested agents while the latter occurs between co-operative agents or non-antagonistic agents. Agents can differ in terms of their roles; they may be active, passive, or both, allowing them to act as a master, slave or peer respectively (Weiss, 2013). Although there are different roles, agents should be able to send and receive information to do their tasks.

3.5.1 Message Types

According to Weiss (2013), there are two basic message types between agents: assertions and queries. Despite agents having different roles, they must have the ability to receive information. Active agents have the ability to issue queries and make assertions as well. With this potential, they can control other agents in the environment. On the other hand, passive agents are able to accept queries from other agents or sources and send a reply to these sources. Finally, peer agents are able to accept and make both queries and assertions. Agent capabilities are shown in Table 3.1.

TABLE 3.1: Agent Capabilities

(Weiss, 2013)

Role	Passive agent	Active agent	Peer agent
Receives assertions	×	×	×
Receives queries	×		×
Sends assertions	×	×	×
Sends queries		×	×

3.5.2 Communication Languages

A multi-agent system is built using several individual agents that work together to achieve the final design objective of the system. The micro level involves the design of an individual agent whereas the macro level involves the design of a multi-agent system in which agents communicate with each other (Wooldridge, 2009). In order to design a multi-agent system, there are two fundamental issues that should be taken into consideration, namely, the communication language between agents and ontologies.

3.5.2.1 Speech Acts

Computational agents communicate with each other using spoken human communication. This type of communication is analysed based on speech act theory in which human natural language is seen as actions such as requests, suggestions and replies (Austin, 1962; Weiss, 2013). According to Weiss (2013), a speech act consists of three aspects, namely, locution (the physical utterance by the speaker), illocution (the intended meaning of the utterance by the speaker) and perlocution (the action that results from the locution). Communication between humans may lead to an equivocal understanding, which as a result may create uncertainties in regard to performing suitable actions or misunderstanding the intent of the message. However, communication among agents should be identified in such a way that agents have no doubt about the type of message.

In order to identify the illocutionary force of utterance, speech act theory uses performative verbs such as promise, report, convince, insist, tell, request, and demand. Weiss (2013) has stated that "speech act theory helps define the

type of message by using the concept of the illocutionary force, which constrains the semantics of the communication act itself". In other words, the sender's intended communication act should be clearly delineated, and the receiver should have no doubt about the type of message sent. Speech act theory has directly influenced the development of a number of languages for agent communication (Wooldridge, 2009).

3.5.2.2 Knowledge Query and Manipulation Language (KQML)

The knowledge query and manipulation language (KQML) is defined as a protocol for exchanging information and knowledge among agents (Weiss, 2013). In order to realise the content of the message, all of the required information should be embedded in the communication itself. The KQML can be seen as an 'envelope' format for messages (Wooldridge, 2009). The KQML structure is presented in Figure 3.2.

```
(KQML-performative
                    :sender      <word>
                    :receiver    <word>
                    :language    <word>
                    :ontology    <word>
                    :content     <expression>
                    ...)
```

FIGURE 3.2: The KQML structure

(Weiss, 2013)

There are two basic components of a KQML message, namely, a performative and a number of parameters. The parameters, which are: *:content* (the

message content), *:ontology* (the terminology used in the message), *:language* (the language that expresses the content), *:receiver* (the intended recipient of the message) and *:sender* (the sender of the message), can be seen as attributes of the message. However, there are other parameters that can be included in the message such as: *:reply-with* (if the sender expects a reply) and *:in-reply-to* (reference to the *:reply-with* parameter) (Wooldridge, 2009). Although the KQML was significant for developing multi-agent systems, many aspects of it were criticised such as the unstandardised transport mechanisms for the KQML messages and the inadequate KQML performative set. As a result, another language was established by the Foundation for Intelligent Physical Agents (FIPA) consortium.

3.5.2.3 FIPA-Agent Communication Language (FIPA -ACL)

FIPA, a non-profit consortium, has developed a new standard relating to the development of agents. The objective was to create standards that would be usable across a vast number of applications (Bellifemine et al., 2007). Based on FIPA's work, the main outcome was the development of an agent communication language (ACL). FIPA-ACL is based on speech act theory, where messages simulate actions (also named speech acts or performatives).

According to Bellifemine et al. (2007), the most common acts are inform, request, agree, not understood, and refuse. The structure of FIPA-ACL and KQML messages is similar, and the message parameters are also similar (Wooldridge, 2009). However, the mandatory part of a FIPA-ACL message is the performative part, which reflects the type of ACL message.

3.5.2.4 Ontology

In order for agents to communicate effectively about a specific domain, it is vital for them to agree on a specific terminology that exactly defines that domain. According to Wooldridge (2009), an ontology is defined as "a specification of a set of terms, intended to provide a common basis of understanding about some domain". It provides the vocabulary of a specific domain (i.e. something is a subset of something) in order to ease the communication between agents in addition to making this communication effective. An ontology is defined as a set of objects and concepts in which the relationships between these objects are clearly defined and described (Weiss, 2013). Moreover, the classes and relationships should be identified in the ontology without their instances. In other words, an ontology is more than a taxonomy of classes (Weiss, 2013). For the development of agents, developers must use a specific ontology that reflects the agents' knowledge. The ontology may be seen as a tree of concepts that forms a hierarchical structure.

3.6 Agent-Oriented Analysis and Design

The system analysis and design phase is a paramount process in the System Development Life Cycle (SDLC). It is used to understand the requirements of any system and achieve its design objectives. Models that describe the system are created and designed in this process. These models are tentative and rather abstract especially in the early stages of this process. However, they become increasingly more concrete and detailed closer to implementation (Wooldridge, 2009).

According to Wooldridge (2009), methodologies for the analysis and design of agent-based systems are classified into two categories, namely, extended object-oriented development for the purpose of agent-oriented software engineering (AOSE) or adapted knowledge engineering methods.

3.6.1 The AAII Methodology

The Australian AI Institute (AAII) methodology for agent-oriented analysis and design was developed as a result of building applications such as the distributed multi-agent reasoning system (DMARS). It is based on object-oriented methodologies incorporating agent concepts (Wooldridge, 2009). The aim is to construct a set of models that is fully elaborated and used to define an agent system specification. AAII offers both internal and external models. The internal model deals with the internals of agents such as their beliefs, desires, and intentions. On the other hand, the external model is concerned with the agents themselves and the relationship between them.

3.6.2 Gaia

Wooldridge et al. (2000) have developed Gaia as a methodology for agent-oriented analysis and design. This methodology is generic and comprehensive; it is suitable for various multi-agent systems and deals with both the micro and macro level of systems. The use of this methodology allows analysts to start systematically from the requirements to a sufficiently detailed design in which the system can be implemented directly (Wooldridge et al., 2000). The motivation behind this methodology was that previous methodologies failed to describe agents' characteristics such as autonomy and problem-solving. Gaia

can encourage developers to think of building agent-based systems as a process of organisational design (Wooldridge, 2009). The concepts of Gaia fit into two categories, namely, abstract (entities are used in the analysis phase to conceptualise the system) and concrete (entities are used in the design phase and have their instances at runtime).

In general, the objective of the analysis phase is to get an efficient understanding of the system and its structure. However, in Gaia, this understanding is captured in the system's organisation (Wooldridge, 2009). The system's organisation involves a collection of roles with relationships between them. Therefore, the first step in the Gaia analysis involves role setting in the system, and the second deals with the modelling of the relationships among the roles. According to Wooldridge (2009), a role can be determined using four attributes, namely, responsibilities, permissions, activities, and protocols. The responsibilities associated with a role are divided into two types: liveness and safety properties. Liveness properties are used to provide a positive impact to the system. In contrast, safety properties are used to ensure that there is nothing critical happening to the system. Permissions can be defined as the privileges that the role can have in order to perform a task and to ensure that the role is permitted to do it. Activities are the computational tasks that a role can perform without the intervention of other roles. Finally, protocols define the way a role can interact with other roles.

3.6.3 Tropos

The Tropos methodology aims to develop agent-oriented systems throughout the software development lifecycle (Bresciani et al., 2004; Giorgini et al., 2005). Tropos is based on two vital ideas. First, the notions of agents such as goals and

plans are used in all phases of the system development life cycle, from the analysis phase to the physical application of the system. Second, Tropos involves critical requirements analysis in the early stage of the analysis in which the analyst can gain an insightful understanding of the agents' environment and the kind of interactions between the agents. According to Bresciani et al. (2004), the Tropos language is used for conceptual modelling using Unified Modeling Language (UML) class diagrams.

The first stage of analysis in Tropos consists of developing two models: an actor model and a dependency model (Wooldridge, 2009). The actor model depicts the main actors (stakeholders) in the system in terms of their strategic interests (goals). The dependency model is used to document the dependencies between these actors. Then, a goal model is formulated and associated with a plan model to achieve the desired goals.

3.6.4 Prometheus

The Prometheus methodology involves three primary phases: system specification, architectural design and detailed design (Padgham and Winikoff, 2002). The system specification phase involves the process of identifying the goals and functionalities of the system. In addition, it defines the interface in terms of inputs (percepts) and outputs (actions) between the system and its environment. The architectural design phase refers to the process of identifying the actual system structure where different agents interact with each other. The detailed design phase is concerned with the structure and the capabilities of each agent.

3.7 Agent Platforms

Agent platforms can be seen as the environment for building and modelling agent-based systems in which agents can effectively communicate and operate with each other to achieve the overall design objectives. There are major implementations of agent platforms that comply with the FIPA specifications (Foundation for Intelligent Physical Agents, 2016). These platforms are discussed in more detail in the following sub-sections.

3.7.1 JADE Platform

JADE is an open source java-based platform for building agent-based distributed systems under the Library Gnu Public Licence (LGPL) by Telecom Italia (Bellifemine et al., 2007). JADE has many features that simplify the realisation of distributed systems that use the software agent abstraction (Wooldridge and Jennings, 1995). This abstraction is implemented using a well-known object-oriented language called Java, in which a simple and friendly Application Program Interface (API) is provided. Moreover, the JADE platform can be fully distributed in many hosts in a fixed network. According to Bellifemine et al. (2007), JADE platform has many useful functionalities that facilitate the development of agent-based applications; it is a fully distributed system (i.e. each agent can run as a separate thread in different locations) and is compliant with the FIPA specifications. Furthermore, it provides a simple and effective agent life-cycle management in addition to a set of graphical tools that helps the developer in debugging and monitoring the agent life cycle. It is also suitable for limited-capability devices such as mobiles (Kravari and Bassiliades, 2015).

The JADE platform is composed of agent containers where the agents live. These containers can be distributed across the network. However, there is a

special container in the JADE platform called the main container (Bellifemine et al., 2007). This container is the focal point of the platform. Moreover, it is the initial container to be launched and the other containers should register with it. The UML diagram, which presents the main architectural elements of JADE, is shown in Figure 3.3.

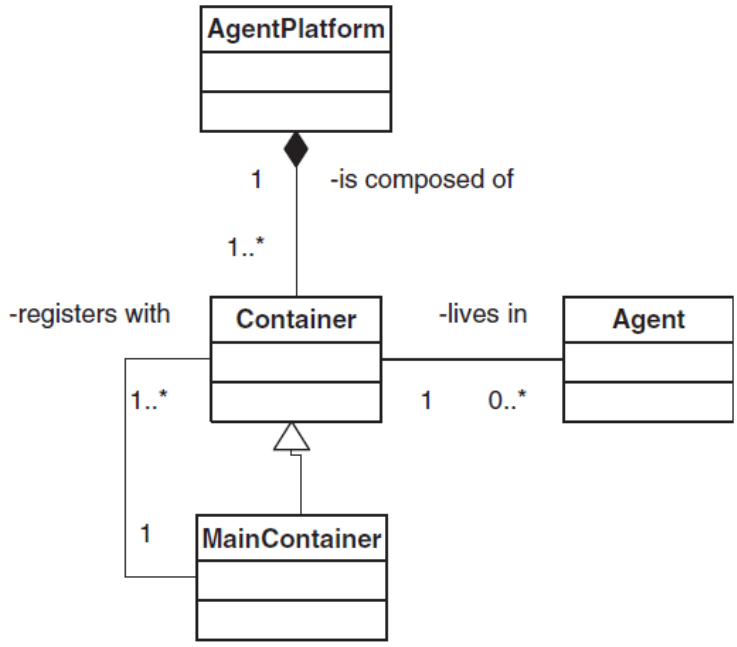


FIGURE 3.3: The main architectural elements of JADE (Bellifemine et al., 2007)

According to Bellifemine et al. (2007), the main container has three responsibilities. First, it is responsible for the management of the container table (CT), which holds the object references and transport addresses of the other containers. Second, it is in charge of managing the global agent descriptor table (GADT); this table contains information about the agents in the platform such as their current state and location. Finally, there are two special agents that are encapsulated in the main container, namely, the Agent Management System (AMS) and the Directory Facilitator (DF). The AMS agent is designed to

provide a naming service in which each agent has a unique name. In addition, it presents the authority in the system; it can kill or create agents on remote containers and it provides the white-page service (Bellifemine et al., 2007). The DF agent is designed to provide yellow-page services whereby an agent can find the services it requires from other agents to achieve its goals.

JADE is one of the most common platforms that researchers use for building multi-agent systems. A study by Kravari and Bassiliades (2015) concludes that JADE is currently the most common FIPA-compliant agent platform in academia and industry.

3.7.2 JAS (Java Agent Services API)

JAS is an open source agent platform that uses agent-based simulation modelling. It was designed by researchers at the Santa Fe Institute. The main feature of JAS is its ability to provide simulation that follows the standard discrete-event simulation model (Kravari and Bassiliades, 2015). According to the Foundation for Intelligent Physical Agents (2016), JAS is based on an industry standard specification and API, which are used to develop agent platform-service infrastructures. Moreover, it is an implementation of the FIPA Abstract Architecture within the Java Community Process initiative and is used for building commercial FIPA compliant systems. It allows the integration of third party plug-ins.

3.7.3 JACK Intelligent Agents

JACK is an agent-oriented development commercial platform built using the Java programming language by Agent Oriented Software Pty. Ltd. (AOS). The JACK Agent Language (JAL) and Extensible Markup Language (XML) are the

main programming languages for the development of agents in JACK (Kravari and Bassiliades, 2015). JACK has been designed to develop agents in complex and dynamic environments. Moreover, the agents in JACK are developed based on the Belief-Desire-Intention (BDI) agent metaphor; these agents have beliefs and goals, and they interact with the environment and other agents in order to achieve these goals. BDI is an intuitive abstraction that allows developers to deal with complex problems. JACK can run on portable devices such as PDAs as well as high end devices such as mainframes. As JACK is based on the Java foundation, it can be run with multiple threads across multiple CPUs, and can be integrated with third-party plug-ins.

3.7.4 ZEUS

ZEUS is an open source agent platform for developing collaborative multi-agent applications. It was implemented using the Java language by BT Labs. ZEUS has a toolkit that provides developers with the tools required to develop multi-agent applications. The architecture of ZEUS has been designed to provide support for generic agent functionality and the planning and scheduling of an agent's actions (Foundation for Intelligent Physical Agents, 2016). In addition, it supports agent communications using FIPA ACL and TCP/IP as the message transport and the delivery mechanism respectively. The agents are built in a visual environment using a sophisticated GUI that enables the developers to redirect agent behaviour. The planning and scheduling approach in ZEUS is based on the idea of representing goals and actions by defining descriptions that contain the necessary resources and pre-conditions that agents require to operate properly (Foundation for Intelligent Physical Agents, 2016). This provides a representation of the goals using an action chain (using a process of backwards chaining) that must be done before the goal can be met.

3.7.5 Lightweight Extensible Agent Platform (LEAP)

LEAP stands for Lightweight Extensible Agent Platform (IST-1999-10211) (Foundation for Intelligent Physical Agents, 2016). This is a platform for developing intelligent agents, and it is the root of the second generation of FIPA compliant platforms. LEAP uses the functionalities of both ZEUS and JADE. The ZEUS environment is used to develop agent applications whereas the JADE environment is used as a run-time environment for these applications. Therefore, LEAP reaps the advantages of the advanced design-time features of ZEUS in addition to the lightweight and extensible features of JADE.

3.7.6 Agent Development Kit (ADK)

The Agent Development Kit (ADK) is a mobile Java-based and FIPA-compliant development platform that was introduced by the Trillian Company. It was designed in order to create and build reliable and scalable industrial applications using intelligent mobile agents. Unlike conventional applications (i.e. the program runs only in one machine), mobile agent applications can travel and perform their tasks on different machines. Tryllian (2016) states that "[t]he ADK enables developers to communicate with the agent software components that are not necessarily controlled agent world, such as databases, search engines, mail systems, Web servers and J2EE applications". The ADK platform allows developers to easily build and manage large-scale distributed applications regardless of the location, data source or protocol. This enables an adaptive, dynamic response to changes (Foundation for Intelligent Physical Agents, 2016). ADK is a commercial platform that provides a free research licence.

3.7.7 April Agent Platform (AAP)

The April Agent Platform (AAP) is a FIPA-compliant platform used to provide a light and powerful tool for the development of agent-based platforms. It has been developed using the April programming language and the InterAgent Communication platform (Foundation for Intelligent Physical Agents, 2016). One of the primary purposes of AAP is to provide a platform for research and experimentation in agent technology. The AAP is an open source platform under GNU public licence.

3.7.8 Comtec Agent Platform

The Comtec Agent platform is an open source agent development platform under GNU general licence. It is a FIPA-compliant platform for implementing FIPA agent communication, agent management, agent message transport and other applications (Foundation for Intelligent Physical Agents, 2016). The implementation of the FIPA Ontology Service and Agent/Software Integration, which require Semantic Language (SL) for the content, reflects the uniqueness of Comtec.

3.7.9 FIPA-OS

FIPA-OS is a component-based toolkit for developing FIPA compliant agents developed by the Emorphia Company and other contributors (Foundation for Intelligent Physical Agents, 2016). It supports the majority of FIPA experimental specifications. Two versions of FIPA-OS have been released, namely, the Standard FIPA-OS and the Micro FIPA-OS. The former has been designed for general purpose whereas the latter has been designed to be executed on

PDAs and smartphones by the University of Helsinki as part of the IST project, Crumpet (Foundation for Intelligent Physical Agents, 2016).

3.7.10 Grasshopper

Grasshopper is an open source Java-based mobile intelligent agent platform. It is a FIPA-compliant platform that incorporates universal agent standards, namely, the OMG MASIF (Mobile Agent System Interoperability Facilities) (Foundation for Intelligent Physical Agents, 2016). Grasshopper enables the integration of open source plug-ins (third-party plug-ins) to provide the OMG MASIF and FIPA standard interfaces for agent/platform interoperability.

3.7.11 Agent Platforms Comparison

As discussed in the previous section, agent platforms can be seen as the environment for building agent systems with different purposes. However, choosing a suitable platform is vital in order to meet the overall objectives of designing such agent systems. In the literature, many agent platforms are used for building agent-based systems. JADE is one of the most common platforms for building agent-based systems. It is an open source and fully FIPA-compliant platform that provides developers with pre-defined packages that facilitate the design and implementation of agent-based systems.

In their survey of agent platforms, Kravari and Bassiliades (2015) conclude that JADE is the most popular platform used by researchers for developing agent-based systems. In the survey, twenty-four agent platforms are compared based on five discrete criteria, as shown in Table 3.2.

TABLE 3.2: The evaluation criteria of agent platforms

(Kravari and Bassiliades, 2015)

Platform properties	Usability	Security	Pragmatics	Operating ability
✓ Developer ✓ Domain ✓ Latest release ✓ Open source ✓ License	✓ Learnability ✓ Simplicity ✓ Scalability ✓ Standard compliance ✓ Communication	✓ Fairness ✓ Platform security ✓ End-to-end security	✓ Installation ✓ User support ✓ Popularity ✓ Technological maturity ✓ Cost	✓ Performance ✓ Robustness ✓ Programming languages ✓ Operating systems ✓ Stability

In order to achieve the objectives of this study, JADE was chosen to build our multi-agent system, one of the core components of the proposed architecture, to support adaptivity in VLEs, since it is an open source and FIPA-compliant platform, purely designed in JAVA. Moreover, it provides the developer with a tool that simulates the interaction between agents to reach the final goal of the designed system. Besides, it is one of the most common platforms, as indicated in the literature.

In the following section, agent technology in the context of e-learning systems is discussed, outlining related work in the field of adaptive e-learning systems based on learning styles.

3.8 Agents in the Context of E-learning Systems and Related Work

Agent-based systems are widely used in different domains ranging from simple to complex applications. Recently, there has been an interest in implementing agents in the context of e-learning systems for different purposes. The use of

agents is a promising approach in the context of e-learning systems. It is applicable in almost every field of e-learning whether that is adaptive learning, collaborative learning or feedback support (Nedev and Nedeva, 2006). For example, Zhang et al. (2010) have pointed out that the use of agents can effectively support the collaboration processes in e-learning for both teachers and students. Moreover, the use of agents has many advantages in e-learning. According to Bokhari and Ahmad (2013), some of the main advantages of agent technology in the context of e-learning are: it is collaborative (i.e. it supports intelligent interactions between students and teachers), it is adaptive (i.e. it supports adaptivity) and it is intelligent (i.e. the system knows about its users and helps them accordingly). Furthermore, Xu et al. (2014) have used intelligent agents in order to personalise the internal learning mechanism in VLEs. They argue that VLEs can be used to achieve e-learning effectiveness when personalisation is integrated into these systems in which learners' needs are satisfied.

In the literature, agent-based systems have been used to support adaptivity in learning management systems. Adaptivity can take different streams in this context: first, adaptive presentation, which deals with the content of the courses offered in the LMS; and second, adaptive navigation, which deals with the sequence and the order of the presented content. However, providing adaptive content based on learners' learning styles in LMSs is the focus of this thesis. There are a considerable number of proposed frameworks, models and systems in which agent technology is used in order to provide and support adaptivity in learning management systems. Some of the most recent and relevant works in the field will be discussed.

Pitigala Liyanage et al. (2013) have proposed a framework for adaptive LMS that can tailor course material to the learning style of learners. The implementation of their work is applied in Moodle based on the Felder–Silverman

learning styles model (FSLSM). Within the framework, three main modules (agents) are used: the learning style monitoring and learning profile creation agent, adaptive content presentation, and the interface enhancement agent and expert recommendation agent. These agents are designed to facilitate the adaptive functionality in Moodle. Chang and Chen (2012) have built a system structure using agents to provide learners with adaptive content based on the FSLSM. A questionnaire is used and designed based on the FSLSM using the ILS (Index of Learning Styles) method to capture learners' preferred learning styles. Morales-Rodríguez et al. (2012) have proposed a system structure for an intelligent learning management system using an agent to filter learning objects based on learners' learning styles. The architecture is applied in Moodle using different modules, namely, the knowledge module (the database of learning objects and learning styles), the interface module (for adaptive presentation of content), the student module, and the tutor module. Alexandru et al. (2015) have proposed an agent-based architecture to extend the Moodle platform to support adaptivity. The architecture has been designed using a multi-agent module as an independent component that can be integrated in Moodle. However, the integration mechanism, including the process of providing real-time data to the multi-agent module, has not been thoroughly investigated. In addition, there is no evidence that their proposed architecture can be integrated in any VLE. Trikha and Godbole (2016) have developed an adaptive e-learning system in which learners' learning styles and knowledge levels are considered. The proposed system is probably still in the early stages of development and application. Lim et al. (2017) have proposed an approach in order to incorporate adaptivity in LMSs based on learners' learning styles. Adaptivity support has been implemented in Moodle. However, from technological perspective, their proposed approach is unable to support dynamic adaptivity in any type of LMS. Several multi-agent based e-learning systems have been developed to

provide adaptive learning such as F-SMILE (Virvou and Kabassi, 2002), Baghera (Webber et al., 2001), EMASPEL (Ammar et al., 2005) and Allegro (Viccari et al., 2007). These systems are not open source and some of them are domain-dependent.

Some of the frameworks and architectures mentioned above are still in the early stages of development, and have been designed to be implemented in a specific LMS. Moreover, the process of tracking and collecting data about learners in VLEs is not clearly identified in most of them. Some approaches rely on queries that may affect the performance of the system. Besides, the process of changing the pedagogical requirements of adaptation (rules) may require re-structuring of the agent-based systems. In other words, adaptivity support in LMSs should be dynamic, platform independent and conducted in a timely manner with respect to efficiency and effectiveness. Databases are the main repository of most of the current LMSs. Data about learners, including their online activities and personal information, are stored in these databases. However, some of these data can be automatically integrated in LMSs from the main university's Student Information System (SIS). This huge amount of data about learners can be used to better understand the learners and enhance the learning process. From this point of view, online activities (e.g. accessing the system, submitting assignments and accessing course content) can be seen as events stored in the database of LMSs. Therefore, the Event-Condition-Action (ECA) model can be used in this type of environment. The ECA model is widely used in Active Database Management Systems and Workflow Management Systems (Zhi-xue et al., 2012). In the following section, the ECA model is introduced to outline the importance of using this model with agents to support adaptivity in VLEs.

3.9 The Event-Condition-Action (ECA) Model

The ECA model has been used in event-driven systems (Denecke, 2012). This model is a reactive model that responds in a real-time manner to any changes in its environment based on pre-defined rules (policies) and conditions. It is used in various critical systems such as autopilot systems and anti-virus systems. It takes the form *on event - if condition - do actions* (Poulovassilis et al., 2006). It can sense the environment and react accordingly in a timely manner based on a simple rule base and this enhances the modularity and maintainability of the system (Papamarkos et al., 2003; Poulovassilis et al., 2006).

According to Poulovassilis et al. (2006), the ECA model is comprised of three components, namely, the event, condition and action. The event component deals with events that may occur in a particular environment (e.g. a LMS). For example, when a student accesses a course in the LMS or submits a quiz, these activities can be seen as events in the LMS and can trigger pre-defined rules. The condition component can be seen as a decision component to check whether the condition is set to TRUE. If so, the action component responds accordingly (in a timely manner) and some tasks are executed.

The ECA model can be seen as a reactive model; actions are fired in response to events when conditions are met. For example, in a database system, an event can be an update operation on the data; this operation can be an insert or a delete command that affect records in the database. The ECA rules are defined in a single rule base instead of being encoded in different programs, which simplifies the management of these rules. Moreover, they are based on a high-level declarative syntax in which analysis techniques can be flexible and amenable.

3.9.1 The ECA Model and Agents in E-learning Systems

Most of the LMSs are designed with a database where all logs and data about learners and their activities are stored and archived. From this point of view, the ECA model can play a significant role in sensing the e-learning environment using database triggers based on pre-defined pedagogical rules, which can be updated by teachers for any new requirements. However, none of the aforementioned architectures and systems has used the ECA model with agents in LMSs to provide adaptive content based on learners' learning styles. The ECA model is used to control these agents following the pedagogical rules to achieve the final objective of the proposed architecture, which is to support dynamic real-time adaptivity in any LMS. Therefore, a computational model is proposed reflecting a hybrid architecture using agent technology and the ECA model. This architecture is discussed in detail in Chapter 4.

3.10 Summary

In this chapter, agent technology has been introduced and discussed. First, we have defined agent-based systems outlining the taxonomy of agents and their characteristics. Moreover, different types of environments have been discussed where agents may interact and communicate. For the development purposes, a review has been carried out of several development platforms and methodologies used by researchers in order to build agent-based systems. In the second part of this chapter, we have discussed some of the most recent and relevant work in the field. Furthermore, the concept of the ECA model has been introduced and discussed to highlight its significance in e-learning environments and the possibility of integrating this model with an agent-based model.

In the next chapter (Chapter 4), our computational model is proposed, reflecting a hybrid architecture using agent technology and the ECA model in order to support dynamic real-time adaptivity in any LMS based on learners' learning styles.

Chapter 4

Architecture

Objectives:

- To define our computational model
- To describe the proposed architecture and its components
- To highlight the significance of using agents with the ECA model
- To introduce the proposed adaptation process

4.1 Introduction

The first part of this chapter aims to introduce the computational model on which our proposed architecture will be based. The computational model identifies the main computational units of the proposed architecture and the behaviour of these units in order to achieve the desired goal of this thesis. The second part aims to describe the proposed architecture in detail, reflecting a novel approach based on the concept of the ECA model and Intelligent Agents to support adaptivity based on learners' learning styles in any given LMS. Each component of the architecture has a specific role that is vital for the other components. In the following section, the informal computational model is defined and described along with its main computational units.

4.2 Computational Model

A computational model can be defined as a model that describes the computational units of any given system as well as the behaviour and communication between these units. These units can be expressed in terms of entities that interact with each other in a particular system in order to achieve the desired objectives of that system. Thus, the informal computational model on which our work throughout this thesis is based is described below. The computational model describes our hybrid approach including the main computational units and the behaviour and communication between these units. The model is depicted in Figure 4.1.

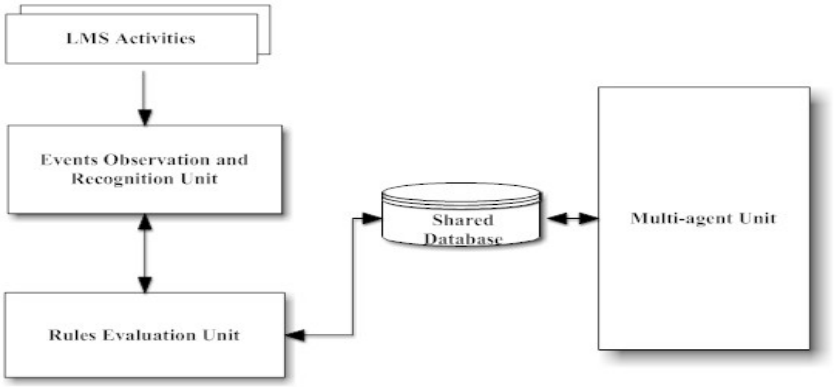


FIGURE 4.1: Computational Model

4.2.1 Units of Computation

The computational model has several units of computation in which each unit has specific roles in order to achieve our design objectives. These units are discussed in the following subsections to identify their roles in the system.

4.2.1.1 LMS Activities

The LMS activities involve all of the learning activities that may occur in the LMS. There are a huge number of logs that are relevant to these activities, which are stored in the database of the LMS. However, the activities that can have a direct impact on the adaptation process proposed in this thesis are considered. Although the adaptation process depends mainly on the completion of the learning styles questionnaire activity (discussed in Section 4.3.2.3), there are other activities that should also be taken into consideration. For example, new enrolment activities in which new learners are enrolled on each course are crucial for the adaptation process. Moreover, quiz attempts are incorporated in order to provide learners with feedback on their progress.

As discussed earlier, in Chapter 2, building a learner model is one of the essential parts of adaptivity. Since adaptive learning systems are designed to meet learners' needs, this model can be considered as the backbone of the adaptation process. In order to achieve this, however, data about the aforementioned activities should be observed and recognised in the LMS. These activities can be considered as events in the database of the LMS. It is very important to point out that only the required events should be observed and recognised. In other words, there should be a mechanism to distinguish between the different types of activities in the LMS. Therefore, the ECA model can play a significant role in capturing and sensing the required events in a timely manner from the database of the LMS.

4.2.1.2 Events Observation and Recognition Unit (EORU)

The EORU has the role of monitoring and recognising the events in the LMS. Any event that is required for the adaptation process is observed and recognised as previously discussed. This unit is one of the most important components of the ECA model. As discussed earlier, in Chapter 3, the ECA model is composed of three components, namely, Event, Condition and Action. The Event component deals with the events that happen in a particular environment. Once the event is recognised and identified, an evaluation process under pre-defined conditions takes place in order to decide whether or not to carry out the actions. Therefore, the time factor is vital in systems that are based on the ECA model. They should respond in a timely manner in most cases. For example, anti-virus systems are designed to protect computers from viruses and malware. When a threat is detected, these systems perform specific actions such as remove, repair etc. Thus, such actions should be performed in a timely manner in order to avoid a catastrophic situation.

The required events for the adaptation process are monitored and recognised by the EORU. Once these events are recognised, the evaluation process is carried out by the Rules Evaluation Unit under pre-defined conditions.

4.2.1.3 Rules Evaluation Unit (REU)

The REU is responsible for the evaluation of the pre-defined rules that are relevant to the adaptation process. It reflects the second and third components of the ECA model where several conditions are evaluated before the required actions are performed. The pre-defined rules can be seen as pedagogical rules that can control the adaptation process. Moreover, these rules can be identified by teachers and course designers in order to guide the adaptation process. The evaluation process in the REU is a significant process in regard to deciding whether or not actions should be performed. Once the rules and conditions are met, the REU provides the Multi-agent Unit with the required event data in a timely manner via the Shared database in order to complete the adaptation process.

4.2.1.4 Multi-Agent Unit (MAU)

The MAU is one of the core components of the computational model. This unit is responsible for supporting adaptivity in the LMS. Moreover, it consists of intelligent agents that work together in order to provide learners with adaptive content based on their learning style. The MAU is designed by taking into consideration the concept of agents, as discussed earlier, in Chapter 3. Furthermore, the adaptation process is based on events that have already been captured by the EORU. Within the MAU, the agents react according to these events and support adaptivity in real time. Although this unit can be seen as a

single unit, it consists of several agents in which each agent has a specific role in its structure. The internal structure of the MAU is discussed later in Section 4.3.4.

4.3 Architecture

In this section, the proposed architecture of the system is explained, reflecting a novel approach in order to support adaptivity in LMSs. Each component of the architecture is discussed in detail, to identify the processes required to achieve our desired goals. The architecture consists of four main components, namely, the Virtual Learning Environment, the ECA Module, the Shared Database and the Multi-agent Module, as shown in Figure 4.2.

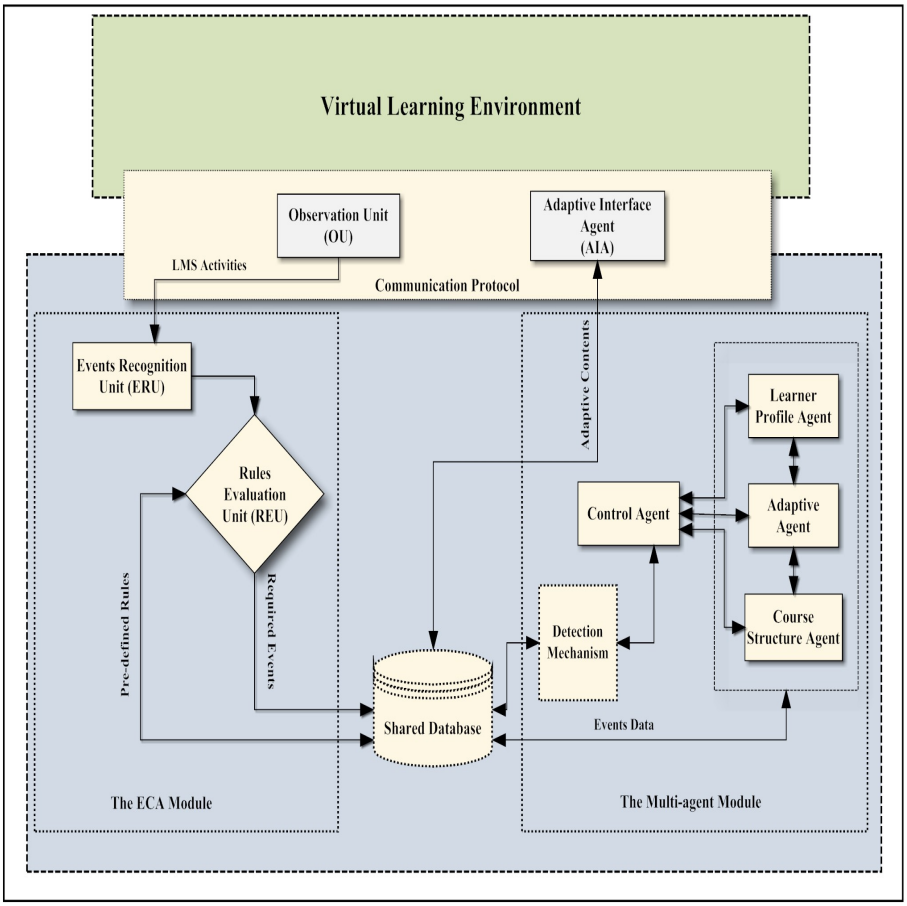


FIGURE 4.2: Architecture (Al-Omari et al., 2016)

4.3.1 Virtual Learning Environment

A VLE is a web-based learning platform where learners and teachers can interact and collaborate in cyberspace. LMSs can be considered as VLEs, as discussed earlier, in Chapter 2. They are designed to facilitate the learning and teaching process. These systems provide powerful tools for both learners and teachers. Moreover, they are designed to store all of the activities performed by the learners in their database. For example, if a learner has accessed a particular course or has completed an assignment, logs about these activities will be stored in the database of the LMS. These activities are considered to be events

occurring in the LMS. The events that are required for the proposed adaptation process were identified earlier in this chapter. The proposed adaptation process depends mainly on the completion of the learning styles questionnaire as a learning activity. This activity is considered to be an event in the LMS. The learning styles capturing process is discussed in the following subsection.

4.3.1.1 Learning Styles Capturing Process

The learning styles capturing process is considered to be one of the main requirements for the adaptation process to take place. The adaptation process starts based on this process; learners are asked to complete the learning styles questionnaire in order that their preferred learning styles can be identified. The questionnaire is always available at the course page level so that learners can submit it whenever it is needed; this provides more flexibility in dealing with different types of courses. The completion of the learning styles questionnaire is crucial for building the learner model, which is involved in the adaptation process. The design of the learning styles questionnaire is discussed in the following section.

4.3.1.2 Felder-Silverman Learning Styles Model (FSLSM)

In the literature, there are several learning styles models and these have been discussed in Chapter 2. However, the adaptation process in our architecture is based on the FSLSM (Felder and Silverman, 1988). The FSLSM was chosen for the reasons mentioned earlier, in Section 2.6.1.6; the FSLSM is one of the most popular and reliable learning styles models in the literature. Moreover, the matching process between the course content and each learning style can

be achieved easily. The Index of Learning Styles (ILS) is basically a questionnaire based on the FSLSM. It was designed and developed by Felder and Soloman (Soloman and Felder, 2005). Furthermore, it is used to capture learners' preferred learning styles. It is used by many researchers as it is a reliable instrument to classify learners based on their preferred learning style. It can be concluded from the literature that capturing learners' learning styles can be achieved using external or internal methods; external methods involve the use of external instruments such as questionnaires and surveys to determine the preferred learning styles of learners. In contrast, internal methods involve the use of data mining and other techniques that may require time in order to better understand the behaviour of learners. These methods are based on tracing and logging learners' activities and behaviour during the learning process. However, in order to meet our objective of providing learners with adaptive content in a timely manner in any LMS, the ILS questionnaire has been adapted as an external method.

4.3.1.3 Learning Styles Questionnaire

The learning styles questionnaire is used to capture learners' learning styles in the LMS. It is based on the Felder-Soloman Index of Learning Styles. The ILS questionnaire consists of 11 forced-choice questions for each of the four dimensions of the FSLSM (a 44-item questionnaire). Each question has two possible responses, "a" or "b", where each response reflects one of the categories in the dimension. A score is obtained - which is an odd number between -11 and +11 - by subtracting the "b" responses from the "a" responses. For example, for the category Activist / Reflective the learners are classified based on their responses to these questions. The classification takes the form of *Strong*, *Moderate* or *Balanced* with respect to the learning style within each category. In other

words, learners are classified into each category depending on the score they receive on the scale, between -11 and +11.

Based on the recommendations by Felder and Soloman, learners are classified depending on specific criteria; a learner who receives an absolute score in a particular category between 1 and 3 has a fairly well-balanced preference in that category. Moreover, a learner whose absolute score in the category is between 5 and 7 has a moderate preference for a specific learning style in that category. The last criterion is for scores between 9 and 11, which indicate that the learner has a strong preference for a specific learning style in the category. An example of the scale is shown in Figure 4.3. The ILS is free for non-commercial purposes and can be accessed online [1]. The implementation of the ILS questionnaire in the LMS is discussed in Chapter 5.

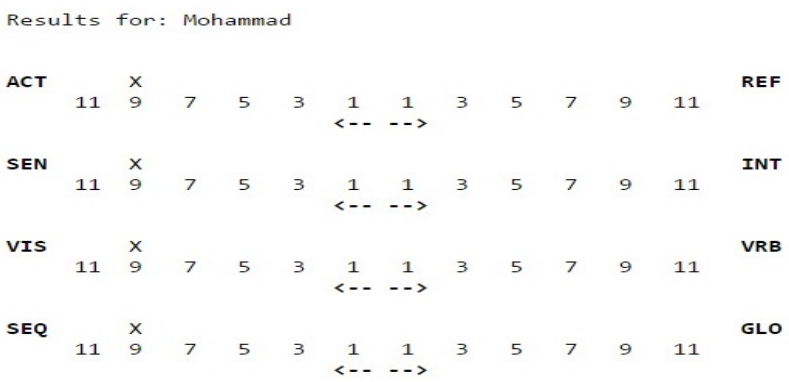


FIGURE 4.3: The scale of ILS

4.3.1.4 Adaptive Course Design and Structure

Courses can be designed in different ways in the LMS depending on the course designers and teachers. However, these courses are designed based on a specific course structure in the LMS; each course is divided into chapters or sections

[1]The ILS can be accessed at: http://www.engr.ncsu.edu/learningstyles/ilsweb.html

in which learning resources and learning activities are presented. In this thesis, the design of an adaptive course is based on the assumption that each course consists of sections and each section reflects a specific chapter. Moreover, the first section in each course is considered as a non-adaptive section where teachers can add any content that is considered to be for all learners. For example, teachers can add announcements, online exams, course descriptions etc. The adaptive sections deal with the adaptive content, which is dedicated to each learner. Therefore, a generic model of an adaptive course structure was designed, as shown in Figure 4.4.

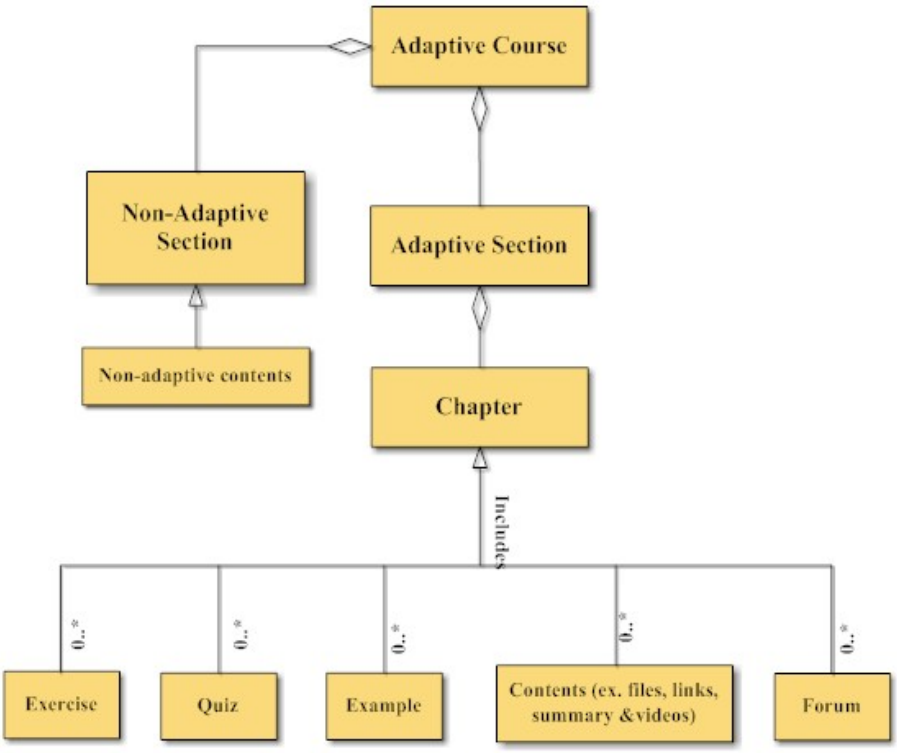


FIGURE 4.4: The generic model of an adaptive course structure

The proposed approach in this thesis can deal with any type of content. The content in the LMS can be generally categorised into two types, namely learning activities and learning resources. The former includes activities with which

learners can interact, such as quizzes, exercises, discussion forums etc. The latter reflects the actual learning files and resources that are used in the learning process in order to understand a specific subject, such as videos, document files, links etc. Most of the current LMSs enable teachers and course designers to include almost any type of content. Furthermore, hash values for repeated content are used in order to save storage space in these systems. Thus, the adaptation process in our architecture can deal with any type of content that can be presented in the LMS.

4.3.1.5 Adaptive Features

The focus of this study is to support adaptivity in LMSs based on learners' learning styles. The adaptation process in the proposed architecture aims to provide learners with adaptive content that suits their preferred learning style. This content is shown in the LMS using the Adaptive Interface Agent. However, different adaptive features can be incorporated in this process. There are three main adaptive features that are integrated in the proposed adaptation process in order to support adaptivity, namely the type, the number, and the order of the course content that should be presented to each learner. In order to achieve this, these adaptive features must comply with each learning style in the FSLSM. The adaptive features are discussed, according to the FSLSM, below:

- **Active/Reflective**

  Active learners are fully motivated to work within groups. They tend to discuss things with others and explain things to them. Moreover, active learners prefer to learn by actively trying things out. Thus, learning activities such as exercises, quizzes, and discussion forums can help them in

the learning process. In addition, the number of activities is increased for this type of learner. These learning activities are shown before the learning resources. Since examples can be considered as work accomplished by others, active learners tend to be less interested in examples. As a result, the number of examples is reduced for this type of learner. On the other hand, reflective learners are less active in terms of their behaviour than active learners. They tend to learn by thinking about the learning material thoroughly and working individually. Therefore, the number of learning activities, such as exercises and forums, is reduced. Since reflective learners prefer to reflect on the learning material at the beginning of the learning process, exercises and examples are shown after the learning resources (i.e. document files, links, videos, and etc.).

- **Sensing/Intuitive**

  Sensing learners prefer to learn from facts that have apparent connections to the real world. Moreover, sensing learners prefer practical work that can reflect the theoretical part of the subject. They like problem-solving activities as well. From this point of view, learning activities such as examples, exercises and quizzes can support these learners during the learning process. Furthermore, the number of learning activities is increased for this type of learner. It is recommended that examples are shown 'before' the learning resources whereas exercises and quizzes are shown 'after' the learning resources. In contrast, intuitive learners prefer challenging work that tests their knowledge. Thus, the number of quizzes is increased and these quizzes are shown before the learning resources. Since intuitive learners dislike repetition, it is recommended to decrease the number of examples and the examples are shown after the learning resources.

- **Visual/Verbal**

  Visual learners prefer visual content in order to better understand a particular topic. It is recommended that the content should include several images, figures, charts and videos. Visual learners memorise better when they see objects and as a result can enhance their learning process. Therefore, learning resources that include visual content are presented to these learners. On the other hand, verbal learners prefer textual content in addition to auditory content. Visual content can be distracting and confusing for this type of learner. Therefore, these learners are provided with textual content that suits their preference. In addition, recorded lectures are probably beneficial for this type of learner.

- **Sequential/Global**

  Sequential learners prefer to learn in a sequential manner focusing on details and connections between the learning topics. It is recommended that learning activities such as examples, exercises and quizzes are presented after the learning resources for this type of learner. In contrast, global learners prefer to learn randomly without considering the connections between subjects. Therefore, it is recommended that this type of learner be provided with outlines before the learning resources in order to help them understand the big picture around the learning material. Moreover, it is recommended that examples, exercises and quizzes be shown after the learning material so that these learners can gain better knowledge after completing the learning material.

At this stage, it is very important to mention that the adaptive features mentioned above are considered as 'recommendations' not a 'prescription' for

presenting the content of an adaptive course in the proposed architecture. However, course designers and teachers should take into consideration the concept of learning styles when they design such adaptive courses. Moreover, other learning activities can be incorporated in order to support a specific learning style. The requirements for course designers and teachers are discussed in the following section.

4.3.1.6 Requirements for Course Designers and Teachers

The adaptive features that are incorporated into the proposed architecture have been discussed in the previous section. However, these features are based on a proposed concept for an adaptive course in which suitable learning activities and resources are presented to each learner. Although the proposed concept can present suitable content based on learners' learning styles, it is crucial that course designers and teachers are involved in the process of designing such adaptive courses. They are required to select suitable learning activities and resources that can be supportive for each learner based on his/her learning styles. In order to achieve this, these learning styles must be taken into consideration by course designers and teachers, who are responsible for designing such adaptive courses. This may require more knowledge about different learning styles and techniques in order to match learning styles with the content. Although this thesis proposes a mechanism for structuring and matching course content with the learning styles of learners, the goal is to develop a platform to support dynamic real-time adaptivity in virtual learning environments.

Most of the current LMSs can provide course designers and teachers with the required tools and facilities in order to design courses in these systems; course designers and teachers can easily add almost any type of learning resources and activities in the LMS. For example, Moodle provides drag and drop

facilities for course designers and teachers in order for them to arrange and manage the learning activities. However, special tools are required for course designers and teachers in order to help them in matching these resources and activities with learning styles. In order to apply the previously discussed adaptive features, our proposed architecture should provide course designers and teachers with the capability to match the course content with learners' learning styles. Moreover, it should provide them with the facility to choose the number and sequence of the activities and resources (i.e. pedagogical rules) that should be presented to the learners. Therefore, a graphical user interface has been designed for this purpose. The interface is based on drag and drop facilities. Course designers can match learning activities and resources created in the LMS with learners' learning styles.

4.3.2 The ECA Module

The Virtual Learning Environment and proposed adaptive features have been discussed in the previous section. An LMS is a platform that is used for designing and presenting learning material to learners. In addition, all of the activities that are performed by learners are stored in the database of the LMS. These activities can be seen as events occurring in the database. From this point of view, the completion of the learning style questionnaire is an event occurring in the LMS that is involved in the adaptation process. Furthermore, the completion of a quiz activity can also be considered an event. Therefore, these types of activities are incorporated into the proposed adaptation process. The ECA module in the proposed architecture is considered as the sensing component; it provides the multi-agent module (discussed in Section 4.3.4) with the required data about LMS events in a timely manner. These data are vital for the adaptation process in order to provide learners with adaptive content. The following

example is given in order to highlight the importance of the ECA module in the proposed architecture: the Air Conditioning System can be seen as a multi-agent system that generally consists of internal and external units; the internal unit is installed indoors and the external unit (the compressor) is installed outside. The internal unit is responsible for informing the external unit whether or not to operate. This process is performed using a sensor, which is installed in the internal unit. The sensor senses the room temperature and sends the required readings to the external unit. Consequently, if the required room temperature is achieved, the external unit will stop operating until new readings are received. From this point of view, the ECA module is considered as the sensing unit in the proposed architecture; it provides the multi-agent module with the required data in a timely manner in order to support adaptivity. The ECA model was discussed previously in Chapter 3. It takes the following syntax:

On Event

Check Conditions

Do Actions

This model is common in database applications, as indicated in the literature. Since databases are one of the main components of any LMS, this model can be applied in these systems and used for many purposes. However, the concept of this model is integrated in the ECA module in the proposed architecture. The ECA module and the multi-agent module operate together reflecting a novel hybrid approach in order to support dynamic real-time adaptivity in any LMS. Database triggers can reflect the concept of the ECA model and are discussed in the following section.

4.3.2.1 Database Triggers and Events

As discussed earlier in this chapter, all logs that are relevant to learners' activities are stored in the database of the LMS. Thus, these logs can be considered as events in which the concept of the ECA model can be applied in terms of database triggers. It is important to point out at this stage that the proposed architecture can be implemented in any given LMS. Therefore, database triggers can be applied in order to sense the LMS and provide the multi-agent module with the required data in a timely manner via the external database, which is completely isolated from the LMS database. This database is named the Shared database.

Database triggers can be defined as internal procedures that are automatically executed when a particular database table is modified (Oracle, 2016). These procedures can be fired at two levels, namely row level and column level. For the row level, the trigger is fired when a particular row in the table is modified. This includes triggering statements such as INSERT, UPDATE and DELETE. These statements are based on Structured Query Language (SQL) and are the most common SQL statements; they can modify the state of the database table. On the other hand, database triggers at column level are fired when a specific column in the table is modified.

The structure of a database trigger consists of the type of event (i.e. INSERT, UPDATE and DELETE) in addition to the scope of the trigger (i.e. row or column level) (Kroenke and Auer, 2012). Moreover, it can include conditions, which should be evaluated before actions are performed. The implementation of database triggers is discussed later in Chapter 5.

4.3.2.2 The ECA Module Components

The ECA module is composed of three parts, namely the Observation Unit (OU), the Events Recognition Unit (ERU) and the Rules Evaluation Unit (REU), as shown previously in Figure 4.2. Each component has a specific role in enabling learners' activities to be sensed in the LMS. Below, each component is discussed in detail.

- **The Observation Unit (OU)**

  The Observation Unit deals with the required locations (Tables), which should be explicitly monitored. This is done to sense the required data, which are essential for the adaptation process. As discussed earlier in this chapter, the adaptation process in the proposed architecture depends mainly on the following events: new course enrolments, the completion of the learning styles questionnaire and the completion of a self-assessment quiz. Therefore, the corresponding database tables are observed and monitored. This can be achieved by applying the concept of database triggers to these tables. However, this component is considered a complementary part of the ERU in which events are recognised.

- **The Events Recognition Unit (ERU)**

  The ERU is the second part of the ECA module. This unit is responsible for detecting any changes that may occur on the tables identified in the OU process. The recognition process in the ERU can be based on any type of transaction in these tables, such as INSERT, UPDATE and DELETE. However, in the proposed approach, INSERT transactions on specific tables can be seen as events in which the type of event can be identified. For example, if a learner is enrolled in a particular course in the LMS, a record

will be inserted into the table of enrolments. The new record indicates that a new learner has been enrolled in that course.

The completion of the learning styles questionnaire can be seen as an event as well. The learning styles questionnaire can be implemented in several ways in the LMS. It can be designed as a quiz, a survey, or a feedback activity depending on the type and the version of the LMS. Survey activities have now been replaced with feedback activities in some of the current LMSs. Therefore, the feedback activity is chosen in order to design the learning styles questionnaire. Since the LMS can include many feedback activities for each course, the ERU should differentiate between the feedback activity that is relevant to the learning style and other feedback activities for the same course. In addition, the ERU should deal with different types of quizzes and differentiate between them, especially for quizzes that are designed to provide feedback to learners based on their score. The feedback is sent via email to the learners.

From this point of view, the Rules Evaluation Unit (REU) plays a major role in evaluating the conditions (rules) that are relevant to the aforementioned activities and is crucial for the adaptation process.

- **The Rules Evaluation Unit (REU)**

  The REU is one of the most important parts of the ECA module. It performs the process of evaluating the pre-defined rules and conditions that control the enforcement of the database triggers; the conditions and rules that are retrieved from the pre-defined rules database are evaluated by the REU. However, if the conditions are satisfied, the required data about the events will be migrated to the Shared database as can be seen in Figure 4.2.

4.3.2.3 Learning Activities Capturing Process

The ECA module is the sensing component in the proposed architecture that provides the multi-agent module with the required real-time data. The required data include learners' responses to the learning styles questionnaire and learners' scores in the self-assessment quiz. Moreover, other related data such as information about learners and their course enrolments should be considered as well. These data are essential for the multi-agent module in order to complete the adaptation process.

The ECA module should sense and track the data as they have a direct impact on the adaptation process in the LMS. Therefore, learners' responses to the learning styles questionnaire should be monitored and stored. The learning styles questionnaire is designed as a feedback activity in the LMS for every course. In the literature, it is mentioned that learning styles may change over time. Thus, the learning styles questionnaire is designed to be shown at the course level. Learners can submit it many times in order to see the adaptive content. This will give them the facility to change the content, especially with other courses. Moreover, learners can choose whether to see all of the content or the adaptive content of the course at any time.

The self-assessment quiz is designed as a quiz learning activity in the LMS and is used to provide learners with feedback via email. Therefore, data that are relevant to this activity are also sensed by the ECA module. The activity diagram for the learning styles and self-assessment capturing process is depicted in Figure 4.5.

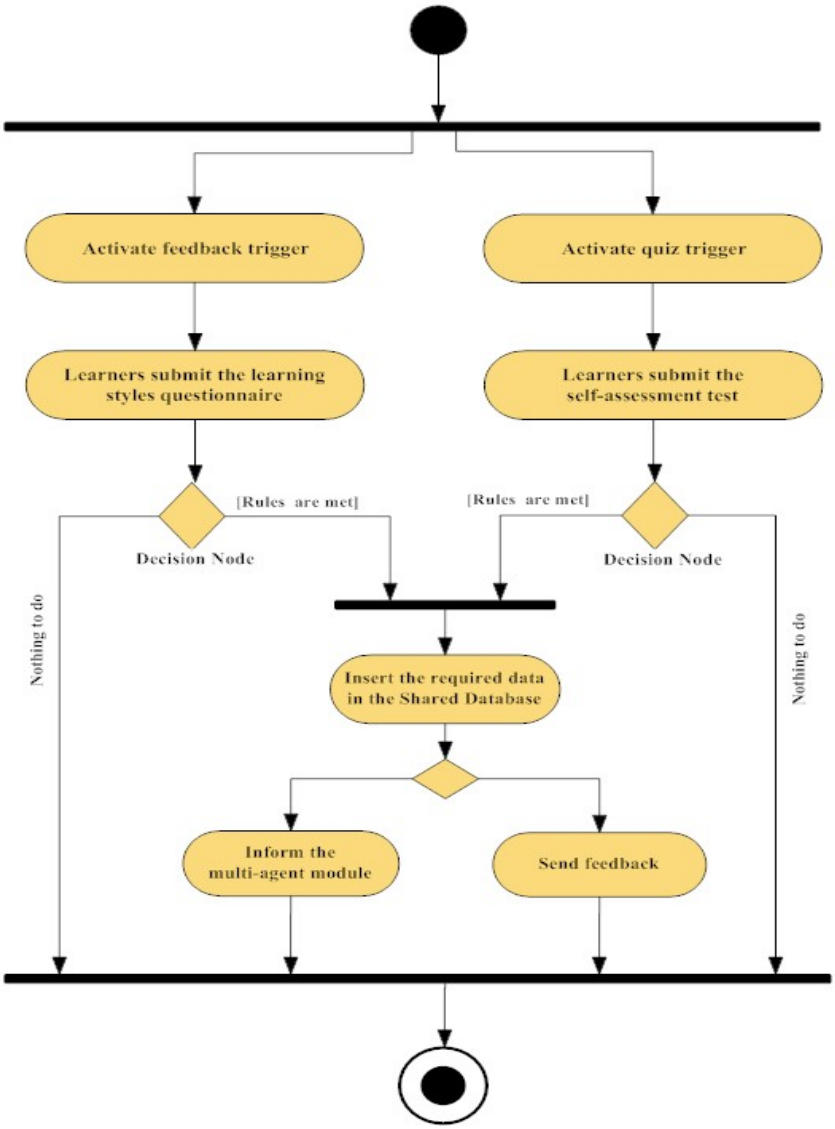


FIGURE 4.5: The activity diagram for the learning styles and self-assessment capturing process

The ECA module deals with all events that are required for the adaptation process. These events should be monitored in the database of the LMS. Although the events discussed earlier are the main events for the adaptation process, there is another event that should be considered as well, namely the new course enrolment event. The process of capturing this event is required in order to collect data about learners and the courses on which they are registered.

These data are used by the multi-agent module to build learners' models. The activity diagram for new course enrolment process is shown in Figure 4.6.

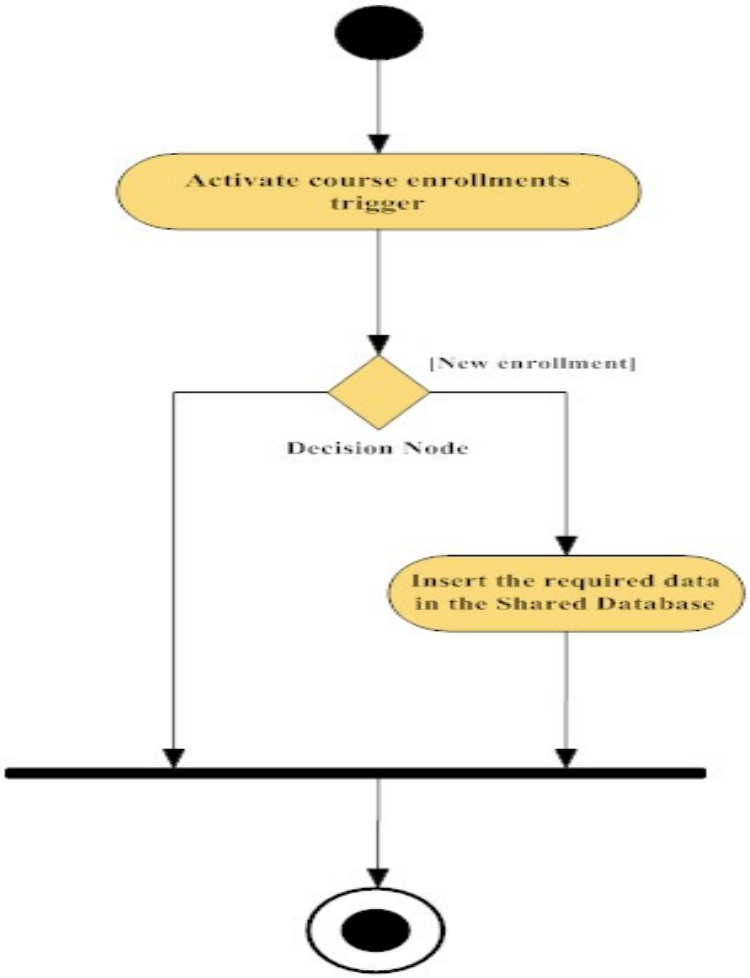


FIGURE 4.6: The activity diagram of the new course enrolment process

4.3.2.4 The Importance of the ECA Module

In the proposed architecture, the ECA module can be seen as the event capturing unit. It is based on the concept of the ECA model. Furthermore, the ECA module is the sensing component of the proposed architecture, which provides the multi-agent module with the required data about LMS events via the Shared

database. The shared database can be seen as the events repository; it is vital for the proposed adaptation process. In addition, this database is designed based on the assumption that the proposed architecture can be implemented in any LMS. The design of the Shared database is discussed in more detail in the following section.

4.3.3 The Shared Database Schema Design

The required events, which meet the pre-defined pedagogical rules, are stored in the Shared database using the TCP/IP communication protocol. It has been named "Shared" since it is shared between the ECA and the multi-agent modules, and the LMS. Data about the events are migrated into the Shared database after which the pre-conditions are met. This process is assumed to be the action part of the proposed ECA module.

The Shared database is a normalised relational database that consists of the database tables that are required for the proposed adaptation process. The schema design of this database is shown in Figure 4.7.

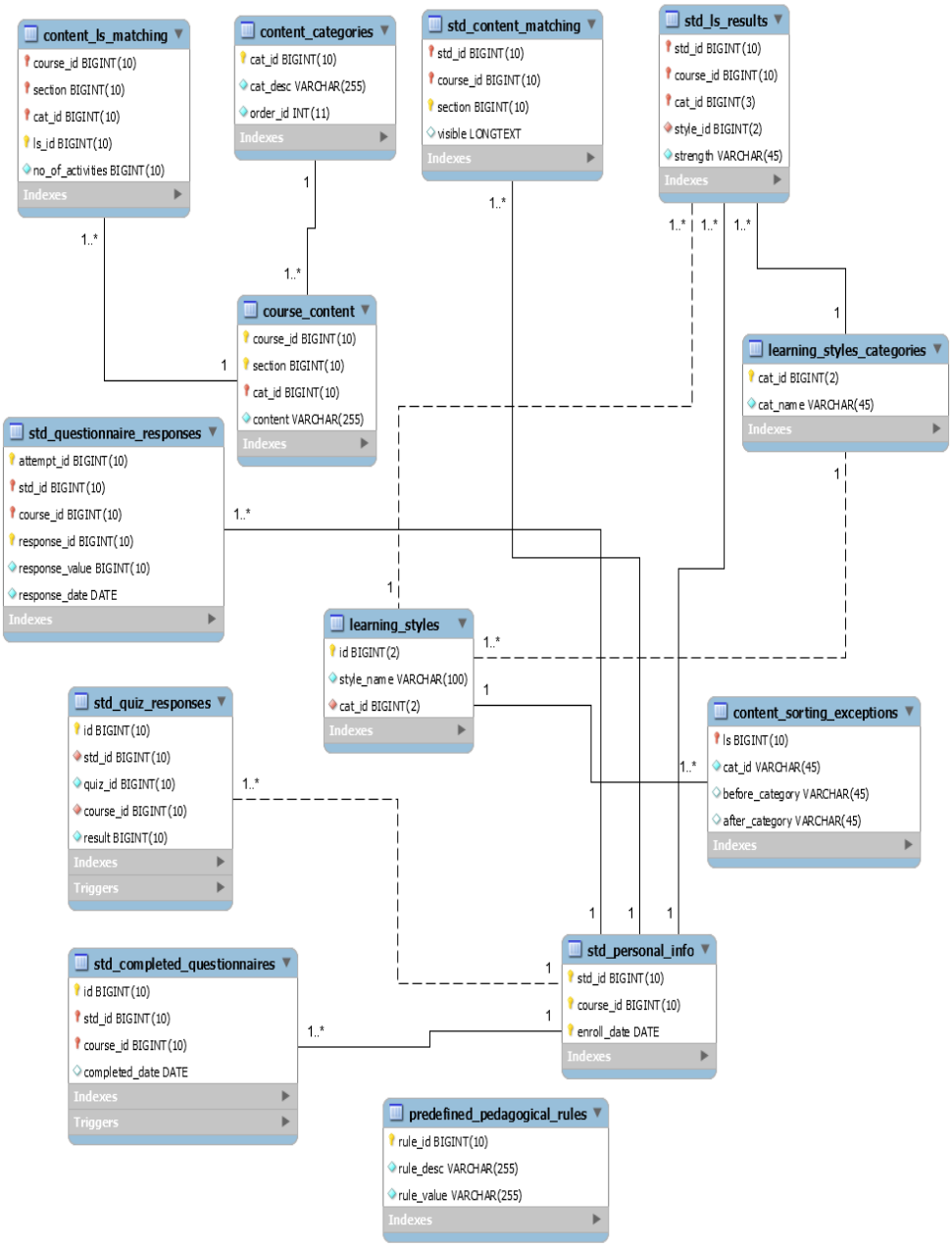


FIGURE 4.7: The schema design of the Shared database

All of the required tables for the adaptation process have been designed as shown in the figure above. The required data about the events are migrated into specific tables based on the type of event. For the main events, which have been highlighted earlier in this chapter, the learning styles questionnaire events are

stored in the table named "std_completed_questionnaire" with the learners' responses stored in "std_questionnaire_responses". The quiz events are migrated into the table named "std_quiz_responses". The description of each table in the Shared database is outlined in Appendix A. The Shared database has also been designed to provide the multi-agent module with the required information about the structure and content of a particular course that supports each learning style. This information is provided by the course designers and teachers via a friendly user-interface, as discussed earlier in this chapter.

The last component of the proposed architecture is the multi-agent module. The concept of intelligent agents is incorporated into the design of this module. The multi-agent module is discussed in detail in the following section.

4.3.4 The Multi-agent Module

In this section, the multi-agent module is discussed to identify the structure of each agent in the module. It is the last component in the proposed architecture. Moreover, the multi-agent module consists of agents that collaborate with each other in order to support adaptivity in the LMS. The concept of an agent is incorporated into the design of this module. There are different techniques in the literature for the analysis and design of agent-oriented systems (see Section 3.6) . However, the Gaia methodology was chosen to design the multi-agent module for the following reasons: First, it is one of the popular methodologies for designing agent-based systems in the literature and has been used by many researchers. Secondly, it provides the facility to design agents as computational components. These components have various interacting roles in order to achieve the final objective of the system. Finally, it meets the requirements for designing the multi-agent module in the proposed architecture.

4.3.4.1 The Gaia Methodology

Gaia is a methodology for the analysis and design of agent-based systems, as discussed earlier in Chapter 3. The Gaia methodology is considered to be both generic - i.e. it is applicable to different types of agent-based systems - and comprehensive - i.e. it deals with the macro (societal) and micro (agent) aspects of a multi-agent system (Wooldridge et al., 2000). By using Gaia methodology, analysts begin with abstract concepts and gradually reach concrete concepts. According to Wooldridge (2009), the analysis and design process can be considered as a process of building increasingly detailed models for the system being developed. There are several models involved in the analysis and design process in Gaia. The main Gaia models and relationships are shown in Figure 4.8.

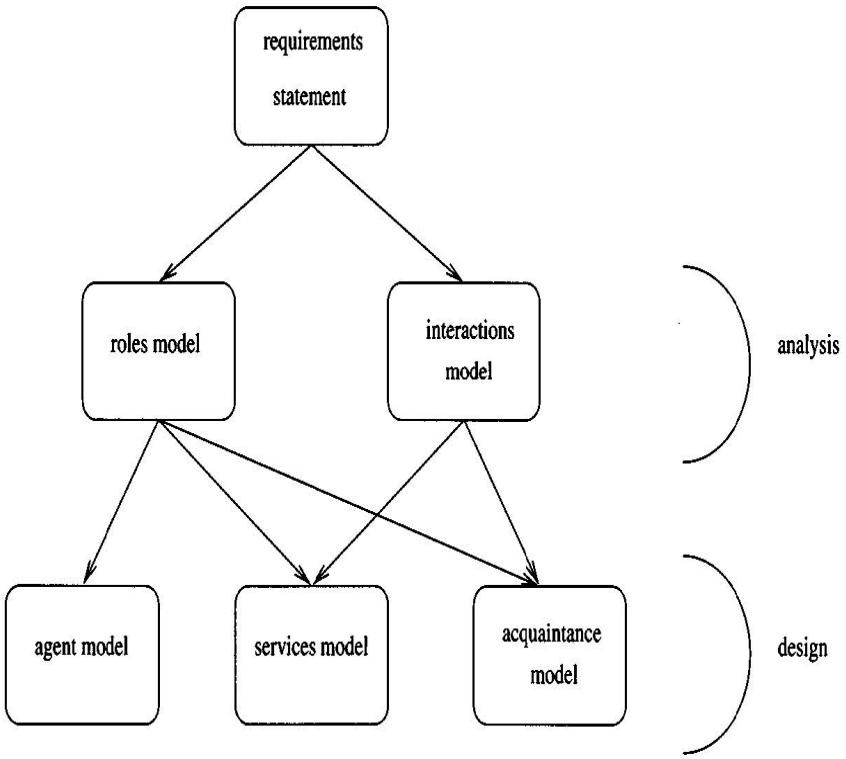


FIGURE 4.8: The main Gaia's models and relationships (Wooldridge et al., 2000)

The analysis stage of Gaia involves the identification of the roles in the multi-agent system and the interaction between these roles in terms of responsibilities and activities. The design stage aims to transform the analysis stage models into a lower level of abstraction so that these models can be easily implemented. The multi-agent module is designed based on the main models of Gaia methodology. In the following section, each agent in the multi-agent module is discussed in detail.

4.3.4.2 The Design of the Multi-agent Module

The multi-agent module consists of several agents, namely the Control Agent (CA), the Learner Profile Agent (LPA), the Adaptive Agent (AA) and the Course Structure Agent (CSA). These agents are designed, incorporating the concept of agents, whereby the agents work together to provide learners with adaptive content based on their learning styles. This content is presented using the Adaptive Interface Agent (AIA). The AIA is excluded from the design of the multi-agent module; it is designed as an interface without considering the concept of agents. The different types of agents and environments have been discussed in detail in Chapter 3. However, the agents in the multi-agent module can be considered reactive and collaborative agents. They react to specific events in the LMS and collaborate together in order to provide learners with adaptive content based on their learning styles. The multi-agent module deals with specific events in the LMS via the Shared database. This database is accessible to the multi-agent module; this module retrieves the required data in order to complete the adaptation process. Therefore, the environment in which the agents operate can be considered an accessible environment. Each agent in the multi-agent module is discussed below.

- **The Control Agent**

The Control Agent is the first agent in our multi-agent module; it is responsible for detecting the relevant events in the learning styles questionnaire in the Shared database. Therefore, the CA is considered the backbone of the multi-agent module; it controls other agents in the system. It receives the required data about events that are relevant to the learning styles questionnaire. These data are provided by the ECA module after fulfilling the pre-defined pedagogical rules, as discussed earlier in this chapter. The CA informs the Learner Agent about these events and passes the corresponding data (ex. learner ID, course ID, event ID, and etc.) to the LA. The role model of the CA is shown in Table 4.1.

TABLE 4.1: The role model of the CA

Role:	ControlAgent (CA)
Description:	This agent is responsible for receiving the required data that are relevant to the learning styles questionnaire events, such as the learner ID, course ID and event ID. It informs the Learner Agent that these events have been detected and provides the LA with the data associated with these events.
Protocols & Activities:	DetectEvent,InformLearnerAgent,SendEventData
Permissions:	Reads events data
Responsibilities:	**Liveness:** CONTROLAGENT=(DetectEvent .InformLearnerAgent .SendEventData)ω[a] **Safety:** Database triggers are activated in the LMS. The Shared database is accessible and running

[a] A cyclic behaviour

It is important to point out at this stage that the CA receives the required data about learning styles events using a detection mechanism in the multi-agent module. This mechanism is based on text files rather than database listeners. Database listeners are external program routines that can periodically monitor a particular database table for any changes. However, these routines may affect the performance of the database of a particular system since they should periodically connect to the database. As a result, this may affect the overall performance of the system. Thus, an approach has been developed so that the detection mechanism is beyond the database context of the proposed architecture. This approach is discussed later in Chapter 5.

- **The Learner Agent**

  The Learner Agent is responsible for collecting the required information about learners and their responses to the learning styles questionnaire. This process is based on the data sent by the CA. Furthermore, the LA has the role of classifying learners based on their preferred learning style. It calculates the strength of the learning styles in the four dimension scale of the FSLSM. This process is adapted from the ILS, which has been discussed earlier in this chapter. The learners are classified based on their preferences as *Strong*, *Moderate*, or *Balanced* in favour of a particular learning style in each of the four dimension scale of the FSLSM.

  The LA stores the data regarding the classification process in the Shared database and provides the Adaptive Agent with the learning styles results for the learners. The role model of the LA is shown in Table 4.2.

TABLE 4.2: The role model of the LA

Role:	LearnerAgent (LA)
Description:	This agent is responsible for classifying learners based on their preferred learning style. It collects the required information about learners and their responses to the learning styles questionnaire based on the data sent by the CA. It provides the Adaptive Agent with the results of the classification and stores these results in the Shared database.
Protocols & Activities:	ReceiveInformMessage,GetLearnerInfo, CalculateLearningStyles,ClassifyLearners, StoreClassResults,SendResults
Permissions:	Reads from the Shared database. Writes to the Shared database
Responsibilities:	**Liveness:**LEARNERAGENT=(ReceiveInformMessage .GetLearnerInfo.CalculateLearningStyles .ClassifyLearners.StoreClassResults .SendResults) ω **Safety:**The Shared database is accessible and running

- **The Adaptive Agent**

  The Adaptive Agent is responsible for the matching process between the content and learners' learning styles based on the classification results. The classification results are received from the LA. Based on these results, the AA can operate in order to match the course content with the learners'

learning styles. The matching process is based on the proposed adaptive features, which have been discussed earlier in this chapter. Furthermore, the AA communicates with the Course Structure Agent to request the course content that matches the learning styles results. Consequently, the CSA provides the AA with suitable content that supports the learner's learning style. Once this content has been received, the AA applies the adaptive features to this content and stores the adaptive content for each learner in the Shared database.

One of the proposed adaptive features is the number of learning resources and activities that should be presented to the learners. This adaptive feature is integrated into the adaptation process so that it can deal with any conflicts that may exist between learning styles. Since learners have different learning styles, conflicts between these learning styles may occur. For example, a learning style may suggest increasing the number of examples for a particular learner, and at the same time another learning style may suggest decreasing the number of examples for that learner. In this case, a moderate number of examples are presented. In order to deal with this situation, weights for the learning style strengths are considered in the proposed adaptation process; the weights for *Strong*, *Moderate* and *Balanced* are (3), (2) and (1) respectively. The AA incorporates these weights into the adaptation process in order to decide on the number of learning resources and activities that should be shown for each learner and to deal with any conflicts that may occur. The course designers and teachers are required to determine whether a particular learning activity or resource supports a particular learning style. In addition, the order of the learning activities and resources that are shown to the learners is also considered. This can be accomplished using drag and drop facilities via

a Graphical User Interface. However, all learning activities and resources that can support particular learning styles are stored in the table "content_ls_matching" in the Shared database. A value of (+1) in the column "no_of_activities" is given for each of these learning resources and activities. Conversely, all learning activities and resources that can affect particular learning styles are stored in the same table with a value of (-1). The learning activities and resources that have no effect on learning styles are excluded. The AA is responsible for ordering the content for the learners based on any criteria identified by the course designers and teachers. The ordering process is designed to be a dynamic process in which the AA can follow any ordering criteria. The role model of AA is shown in Table 4.3.

TABLE 4.3: The role model of AA

Role:	AdaptiveAgent (AA)
Description:	This agent is responsible for the matching process between the content and learners' learning styles based on the classification results. Also, it has the role of ordering the content received from the CA and deciding on the amount of content to be shown to each learner. It applies the proposed adaptive features of the system. After the adaptation process has been completed, the AA stores the adaptive content for each learner in the Shared database and confirms whether or not the mission has been completed successfully.
Protocols & Activities:	ReceiveClassResults,RequestCourseContent, MatchLSContents,CalculateNoActivites, ContentsOrdering,CheckOrderingExceptions ,StoreAdaptiveContents, SendConfirmation
Permissions:	Reads from the Shared database Writes to the Shared database
Responsibilities:	**Liveness:** ADAPTIVEAGENT=(ReceiveClassResults .RequestCourseContent.MatchLSContents .CalculateNoActivites.ContentsOrdering .CheckOrderingExceptions.StoreAdaptiveContents .SendConfirmation) ω **Safety:** The Shared database is accessible and running

- **The Course Structure Agent**

  The Course Structure Agent deals with the structure of the course in the LMS. It collects learning resources and activities that can support and affect the learning styles of learners. This information is crucial for the AA to complete the adaptation process. The role model of the CSA is shown in Table 4.4.

TABLE 4.4: The role model of the CSA

Role:	CourseStructureAgent (CSA)
Description:	This agent is responsible for providing the AA with the course structure including learning resources and activities that can support and affect the learning styles of learners.
Protocols & Activities:	ReceiveClassResults,SendReceiveConfirmation, GetCourseContents,SendContents, ReceiveConfirmation
Permissions:	Reads from the Shared database
Responsibilities:	**Liveness:**COURSESTRUCTUREAGENT= (ReceiveClassResults.SendReceiveConfirmation .GetCourseContents.SendContents .ReceiveConfirmation) ω **Safety:**The Shared database is accessible and running

4.3.4.3 The Agent Model

The agent model is one of the processes of designing the multi-agent module. The agent types and agent instances are identified. The multi-agent module responds to each learning styles questionnaire event captured by the ECA module in the LMS. As a result, each agent in the multi-agent module is instantiated once and has as many instances as the number of events received. The agent model of the multi-agent module is depicted in Figure 4.9.

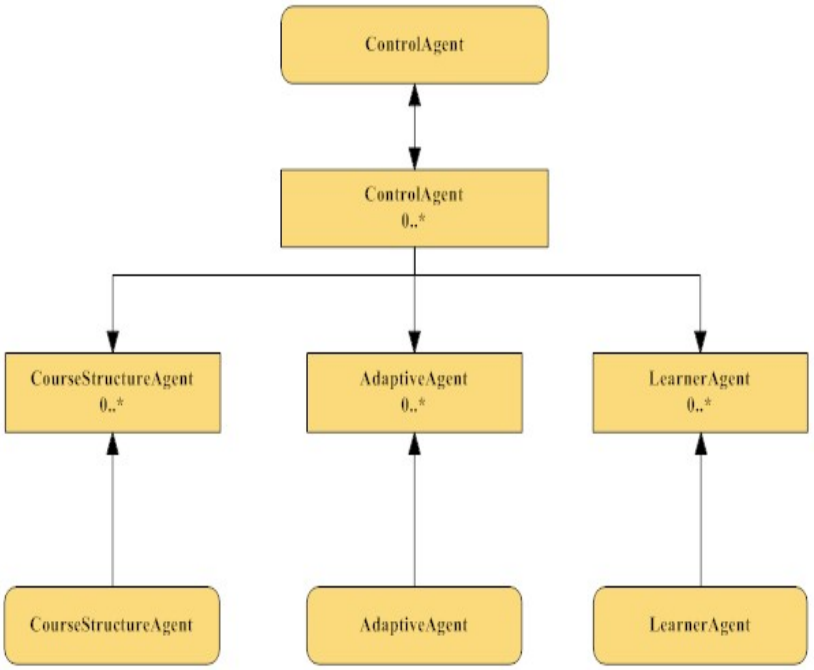


FIGURE 4.9: The agent model of the multi-agent module

4.3.4.4 The Service Model

The service model aims to identify the services that are relevant to each agent role alongside their properties. The properties of these services can be identified in term of inputs, outputs, pre-conditions and post-conditions. The input and output properties can be derived from the activities and protocols that have

already been identified in the role model. On the other hand, the pre-conditions and post-conditions can be identified from the safety property of the role. The main services in the multi-agent module are summarised in Table 4.5.

TABLE 4.5: The main services in the multi-agent module

Service	Detect learning styles questionnaire event.
Inputs	Text file
Outputs	Data about the event (ex. Event ID, learner ID and course ID)
Pre-condition	The multi-agent module is instantiated and running, and the ECA module is activated in the LMS. The learner submits the learning styles questionnaire.
Post-condition	-

Service	Obtain learning styles questionnaire responses.
Inputs	Data about the event (ex. Event ID, learner ID and course ID)
Outputs	Learning styles obtained by classifying the learner based on his/her preferred learning styles
Pre-condition	The multi-agent module is instantiated and running.
Post-condition	The Shared database is accessible using a communication protocol.

Service	Obtain course contents.
Inputs	Course ID and the preferred learning styles
Outputs	The course structure that meets the learning styles preference
Pre-condition	The multi-agent module is instantiated and running.
Post-condition	The Shared database is accessible using a communication protocol.

Service	Adapt contents
Inputs	Learning styles results, course structure and contents
Outputs	Adaptive contents based on learner's preferred learning styles incorporating the proposed adaptive features
Pre-condition	The multi-agent module is instantiated and running.
Post-condition	The Shared database is accessible using a communication protocol.

4.3.4.5 The Acquaintance Model

The acquaintance model is the last model in the Gaia methodology. It represents the communication paths between agents in agent-based systems (Wooldridge et al., 2000). The multi-agent module has four agents, which communicate with each other in order to provide learners with adaptive content based on their preferred learning style. The acquaintance model of the multi-agent module is shown in Figure 4.10. The figure shows an informal model, which reflects the last design phase in Gaia. It describes the abstract interaction of agents in the multi-agent module.

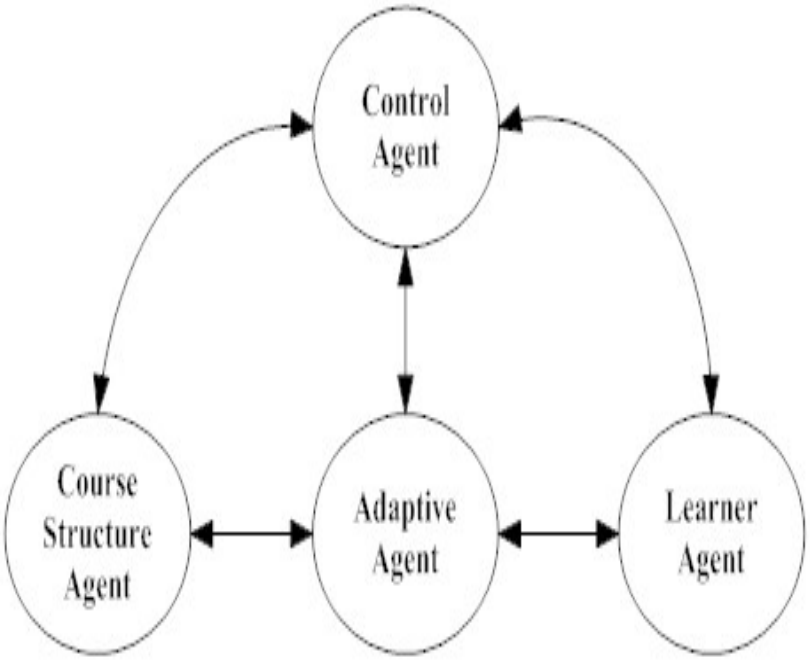


FIGURE 4.10: The acquaintance model of the multi-agent module

4.3.4.6 Communication Language and Ontology

The multi-agent module in the proposed architecture can be seen as a multi-agent system. This system is comprised of individual agents that interact with each other in order to provide adaptive content to the learners in the LMS. This interaction is based on a communication language between these agents and an ontology, as discussed earlier in Chapter 3. The agents within the multi-agent module communicate with each other using the FIPA-Agent Communication Language (FIPA-ACL) (see section 3.5.2). The FIPA-ACL can be seen as communication standards that agents follow when they communicate with each other. In order for agents to communicate effectively about a specific domain, the ontology that describes that domain should be involved in the communication between these agents. Since the multi-agent module deals with the Shared

database, the ontology, including the concepts and classes, is derived from the structure of this database.

Nevertheless, the communication language and ontology depend mainly on the platform used for the development of the multi-agent system. For example, JADE is a FIPA-compliant agent platform used for the development of agent-based systems incorporating the concept of agents. Moreover, it provides useful tools for developing the required ontologies for the multi-agent system. The development of the multi-agent module is discussed later in Chapter 5.

4.4 Discussion

In the previous sections, the proposed architecture was introduced, highlighting all of the components involved in its structure. This architecture has been designed so that it can be implemented in any given LMS in order to support adaptivity in the LMS in real time. Moreover, it reflects a novel hybrid approach based on the concept of the ECA model and Agents. The required events for the proposed adaptation process are sensed by the ECA module and sent to the multi-agent module. The multi-agent module has the role of providing learners with adaptive content based on their learning style. This adaptive content is presented to the learners based on the adaptive features, which have been discussed in this chapter. However, more adaptive features can be applied in the proposed architecture depending on the requirements of the educational institution. Furthermore, the architecture can deal with any type of content that may support the learning styles of the learners. Although the proposed adaptation process can be based on different learning styles models, the FSLSM and its recommendations are integrated into this process for the reasons mentioned in Section 2.6.1.6.

The focus of this study is to support adaptivity in LMSs based on learners' learning styles. As stated earlier in this thesis, a "one-size-fits-all" approach is the basis of most of the current LMSs, and differences between learners are not considered in the design of these systems. However, the proposed architecture aims to support dynamic adaptivity in a timely manner in any given LMS based on learners' learning styles.

4.5 Summary

In this chapter, the computational model has been described. The main units of computation in the model are discussed, and the main role of each unit has been highlighted. Furthermore, the proposed architecture has been introduced based on the computational model. The architecture comprises several components including the LMS, the ECA module, the Shared database and the multi-agent module. The design of each component has been discussed. A proposed structure for an adaptive course in the LMS has been explained in addition to the requirements for course designers and teachers. The adaptive features incorporated in the adaptation process have also been discussed. The architecture reflects a novel hybrid approach based on the concept of the ECA model and Agents in order to provide adaptive content based on learners' learning styles. The prototype of the proposed approach is discussed in Chapter 5.

Chapter 5

System Prototype

Objectives:

- To introduce the technologies and tools used to implement our hybrid approach
- To present a system prototype of the proposed architecture
- To highlight the significance of the proposed approach

5.1 Introduction

This chapter aims to present a system prototype of the proposed architecture, which has been identified in Chapter 4. In addition, it shows how the components of the architecture were developed using a hybrid approach to support dynamic real-time adaptivity in e-learning environments. In order to achieve this, the technologies used to develop the system prototype are introduced and discussed. This includes the concept of database triggers and intelligent agents. The technical aspects, including the design and source code of the system prototype, are included in Appendices A and B.

5.2 Learning Management System: Moodle

LMSs are crucial platforms for the adoption of distance, blended and traditional learning. Moodle and Blackboard are the most common LMSs and are used in many educational institutions throughout the world. Moodle (version 2.8) has been chosen in this thesis as the learning environment in which to implement our system prototype. Moodle is an open source e-learning platform under the General Public License (GNU). It is widely used and probably the most popular open source platform used in educational institutions worldwide (see Section 2.4.1). The proposed architecture depends on events that occur in the LMS, as discussed previously in Chapter 4. One of the primary events is the completion of the learning styles questionnaire. The following section shows how the learning styles questionnaire was designed and developed.

5.2.1 The Learning Styles Questionnaire

The learning styles questionnaire was designed and developed as a feedback activity (i.e. a survey) in Moodle, as can be seen in Figures A.1 and A.2 in Appendix A. It was adapted from the Index of Learning Styles (ILS) questionnaire developed by Soloman and Felder (2005). The questionnaire was developed to be shown at the course level so that learners could complete it at any time to achieve instantaneous adaptivity within different courses. The questionnaire consists of 44 questions in which each question is identified by ID (ex. 1, 2, 3, etc.). Each question has two forced-answers and measures a particular learning style in the four dimensions of the FSLSM (see Figure A.3 in Appendix A). For implementation purposes, the first answer to each question is given a value of (+1) and the second answer is given a value of (-1). These values are stored in the "Shared" database in order to differentiate between learning styles in the same category. Figure 5.1 shows a screenshot of particular questions in the questionnaire.

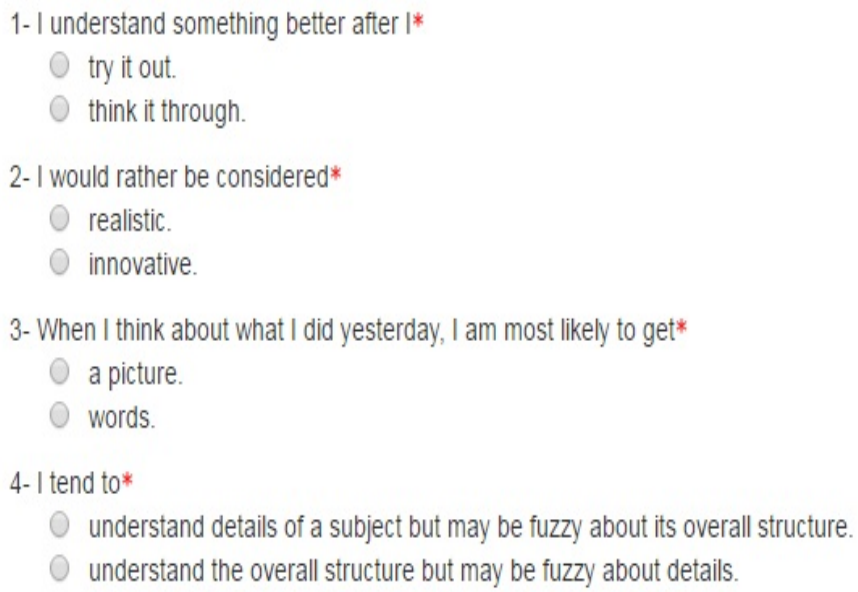


FIGURE 5.1: A screenshot of particular questions relevant to the four dimensions of the FSLSM

As shown in Figure 5.1, each question examines a particular learning style preference in the four dimensions of the FSLSM. For example, the first question inspects the learner's preference in regard to the first category in the FSLSM, which is Activist / Reflective. If the answer "try it out" is chosen, then a value of (+1) is recorded, indicating that the answer supports the Activist learning style. On the other hand, if the second answer (i.e. "think it through") is selected, then a value of (-1) is recorded, indicating that the answer supports the Reflective learning style. Table 5.1 shows all of the questions IDs in the questionnaire and the relevant learning style category.

TABLE 5.1: The questionnaire's questions and relevant learning styles categories

Question No.	**Category**
1, 5, 9, 13, 17, 21, 25, 29, 33, 37, 41	Activist / Reflective
2, 6, 10, 14, 18, 22, 26, 30, 34, 38, 42	Sensitive / Intuitive
3, 7, 11, 15, 19, 23, 27, 31, 35, 39, 43	Visual / Verbal
4, 8, 12, 16, 20, 24, 28, 32, 36, 40, 44	Sequential / Global

As stated before, each question examines the learner's preference in regard to a particular learning style in a particular category depending on his/her response. Each category has 11 relevant questions in order to determine the strength of the preferred learning style. This is determined by calculating the score that the learner achieves in each category. The score takes a value between (+11) and (-11) for each category (i.e. 11 answers), and, as a result, identifies the strengths of the learning style (i.e. *Strong*, *Moderate*, or *Balanced*) (see Section 4.3.1.3). These strengths use integer values as weights. The weights for *Strong*, *Moderate* and *Balanced* are (3), (2) and (1) respectively. These values are used in order to implement the adaptive features and deal with any conflicts that may occur between the learning styles, as discussed before in Section 4.3.4.2

5.2.2 Database Triggers Technique

The required events for the adaptation process should be identified and monitored. In order to monitor and sense these events, the concept of the ECA model is incorporated into the database of the LMS as database triggers. A database trigger is a program stored in the database that is executed by the Database Management System (DBMS) when specific events occur (Kroenke and Auer, 2012). Database triggers are useful in database applications in which real-time

responses are needed. Moreover, they are invoked automatically before or after modifying the data in database tables. However, these triggers might affect the performance of the system depending on the complexity of its design. Database triggers are based on the concept of the ECA model (Poulovassilis et al., 2006; Zhi-xue et al., 2012). In the proposed architecture, database triggers are used to implement the ECA module of the architecture to provide the multi-agent system with the required data via the "Shared" database in a timely manner. Furthermore, they are implemented so that adaptivity can be achieved regardless of the LMS being used. The default DBMS used in Moodle is MySQL (Moodle.org, 2017). However, other DBMSs can be used, such as Oracle and SQL Server (ibid). The syntax of a database trigger in MySQL is depicted in Figure 5.2.

```
CREATE
[DEFINER = { user | CURRENT_USER }]
TRIGGER trigger_name
trigger_time trigger_event
ON tbl_name FOR EACH ROW
trigger_body
trigger_time: { BEFORE | AFTER }
trigger_event: { INSERT | UPDATE | DELETE }
```

FIGURE 5.2: The syntax of a database trigger in MySQL (Oracle, 2016)

The figure above shows the structure of the database trigger in MySQL. The trigger can include several conditions (rules) that should be satisfied in order

to be fired when events occur. These rules can be any rules (ex. pedagogical and management rules) that can guide the adaptation and learning process. The events are recognised based on the type of trigger event. For example, if a particular learner has submitted the learning styles questionnaire, the data for this event are stored in a specific table by means of "INSERT" statements. Thus, a trigger event of type INSERT indicates that the learner has completed the questionnaire for the course. Several conditions (i.e. rules) can be added to the body of the trigger in order to make a decision point regarding whether or not to execute specific statements (actions). The same case can be applied for the other events that are required for the adaptation process. For example, when the learner completes the self-assessment quiz, if the score does not meet the requirements of the course (ex. rules for passing the quiz), feedback will be sent to the learner by email for more support. These events are monitored by implementing the required database triggers in the database of the LMS (see Appendix A). The required events in the LMS for the adaptation process are summarised in Table 5.2.

The purpose of using database triggers in the proposed architecture can be summarised as follows:

- To sense and recognise the events occurring in the LMS in a timely manner subject to pre-defined rules and conditions.
- To provide the multi-agent system with real-time data in order to react accordingly to provide learners with adaptive content.
- To support adaptivity in any LMS regardless of the structure and complexity of its design by integrating the database triggers in the database of the LMS.

TABLE 5.2: A summary of the required events for the adaptation process

Event	Trigger Name	Description and location
New students' enrolment	NEW_ENROLMENTS_TRIGGER	This trigger monitors any new course enrolments. This trigger is installed in the database of the LMS. The relevant data are migrated into the Shared database.
The completion of LS questionnaire	QUESTIONNAIRES_RESPONSES_TRIGGER	This trigger monitors the completion of the LS questionnaire in the LMS and records the learners' responses in the Shared database.
The completion of self-assessment quiz	QUIZ_ATTEMPTS_TRIGGER	This trigger tracks the completion of the self-assessment quiz in the LMS to provide learners with feedback by email.

As stated before, database triggers simulate the ECA module in the proposed architecture. They are crucial as they inform the multi-agent system about any events that are essential for the adaptation process and providing feedback to the learners. In the following section, the multi-agent system is described, reflecting the design of the multi-agent module of the proposed architecture.

5.3 The Multi-agent System

The multi-agent system represents the multi-agent module in the proposed architecture. It is mainly based on the concept of intelligent agents. There are several platforms for the development of multi-agent systems; these have been discussed earlier, in Section 3.7. In this thesis, JADE was chosen to develop

the multi-agent module since it is one of the most popular platforms used by researchers (see Section 3.7.11)

5.3.1 JADE Platform

JADE is an open source platform for the development of agent-based systems using Java as the programming language (Bellifemine et al., 2007). It provides powerful tools that facilitate the development of agent-based systems. It is one of the most common and popular platforms described in the literature (Kravari and Bassiliades, 2015). The interaction between agents can be simulated in JADE to facilitate the validation process during the development phase. The general structure of an agent in JADE is comprised of two main parts: the first part deals with the initiation of an agent and the second part includes the type of behaviour that the agent follows in a particular system. JADE uses FIPA-ACL (see Section 3.5.2.3) as the standard language used between agents to communicate effectively. Therefore, JADE is considered a FIPA-compliant platform that uses several performatives that identify the taxonomy of the agent (see Section 3.7.1).

In the architecture, the multi-agent module is comprised of four agents (see section 4.3.4.2). The Control Agent receives the events provided by the ECA module (i.e. database triggers) using a detection mechanism (discussed later in this chapter). The Learner Agent processes the event data and builds an initial learner model. The Adaptive Agent is responsible for incorporating the adaptive features and providing the required adaptive content to the learner with the cooperation of the Course Structure Agent. Each agent has a specific ID in the system and lives in the main container of the JADE platform. The multi-agent system in JADE is illustrated in Figure 5.3 below.

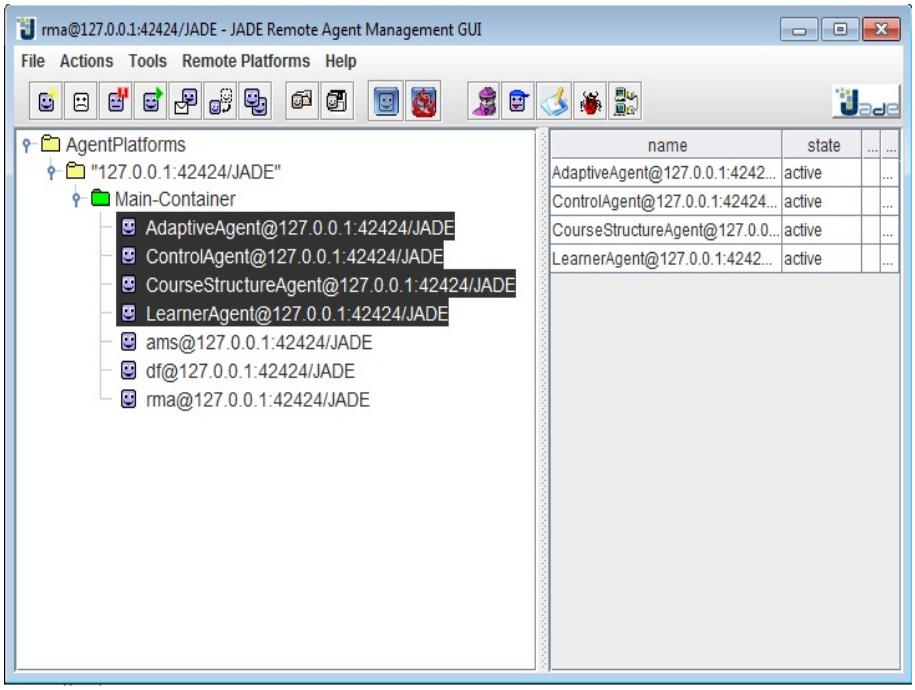


FIGURE 5.3: The multi-agent system in JADE

As shown in Figure 5.3 above, the four agents of the multi-agent system are located in the main container of JADE. The main container is essential to run the agents and other containers if they exist (Bellifemine et al., 2007). It controls the agents and any registered containers. An agent platform in JADE can have several containers other than the main container depending on the requirements of the system. However, any other containers must register with the main container in order to join the platform.

Each agent in the multi-agent system has its own structure depending on its role in the system. The role can be identified by the behaviour of the agent. The multi-agent system is based on two main behaviours, namely simple and cyclic. Simple behaviour refers to simple activities that an agent can perform without repetition (non-cyclic behaviour). In contrast, cyclic behaviour refers to activities that should be repeated continuously depending on specific criteria.

For example, the structure of the Control Agent follows cyclic behaviour; each time the Control Agent receives an event from the ECA module, it should inform and provide the Learner Agent with the required data about that event. The general agent structure is implemented as a *class* in Java, which inherits the properties of an agent class in JADE (Bellifemine et al., 2007). The attributes, activities and protocols for each agent can be represented in terms of a class diagram. The class diagram of the multi-agent system is included in Appendix A.

5.3.2 Events Detection Mechanism

As mentioned before, the ECA module was designed and developed in order to provide the multi-agent system with real-time data about events. In order to achieve this, a detection mechanism was developed so that the database triggers could provide the multi-agent system with the required data about events in terms of 'text' files. The multi-agent system monitors any new text files in a specific folder called the "Events Repository". The action part of the database triggers creates these files in that folder to be detected by the multi-agent system. These files are identified by events IDs in which data about the event are included. The detection mechanism was designed and developed in order to be outside of the context of the database. This may enhance the database performance and the events detection process when compared with *database listeners* (i.e. another technique that can be used). This mechanism was built using Apache Camel.[1] The "Events Repository" folder including events as text files is illustrated in Figure 5.4.

[1] http://camel.apache.org/

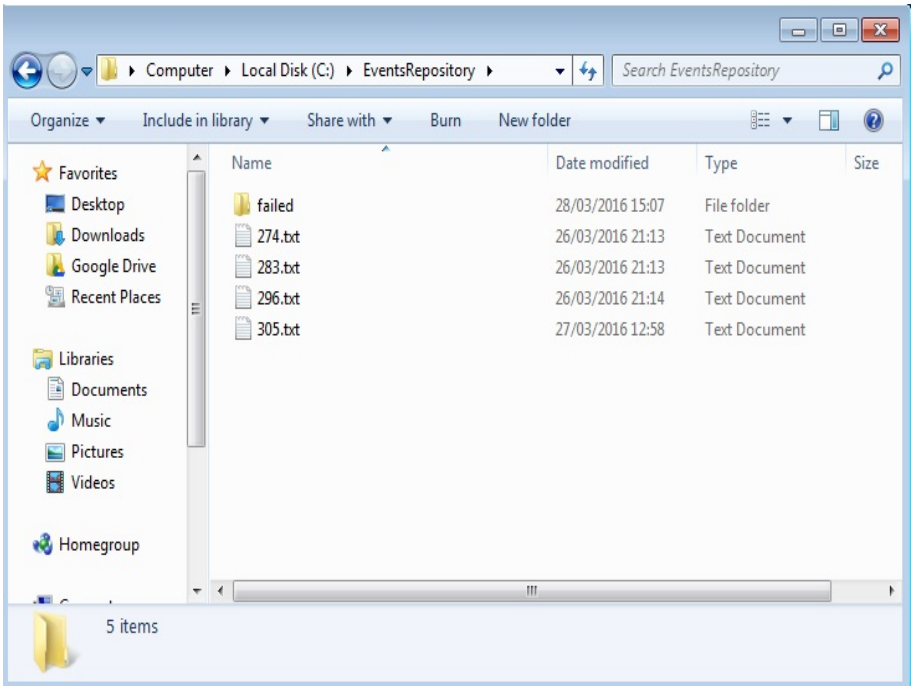


FIGURE 5.4: Events repository as text files

5.3.3 The Multi-agent System Ontology

JADE supports the development of ontologies for agents. The ontology defines the schema of concepts in order to make the interaction between agents effective (Bellifemine et al., 2007). Since the multi-agent system deals directly with the "Shared" database (i.e. data layer), which includes all of the required tables for the adaptation process, the ontologies were developed based on the structure and design of this database. It defines the required concepts that are essential for the agents in order to interact. JADE provides a package called "jade.Concept", which can be used in order to define any concept. A description of each table in the "Shared" database is included in Appendix A.

5.4 Prototype Structure

The prototype structure implements the design of the proposed architecture as shown in Figure 5.5. It provides a hybrid approach using agents and the concept of the ECA model to support adaptivity in the e-learning environment.

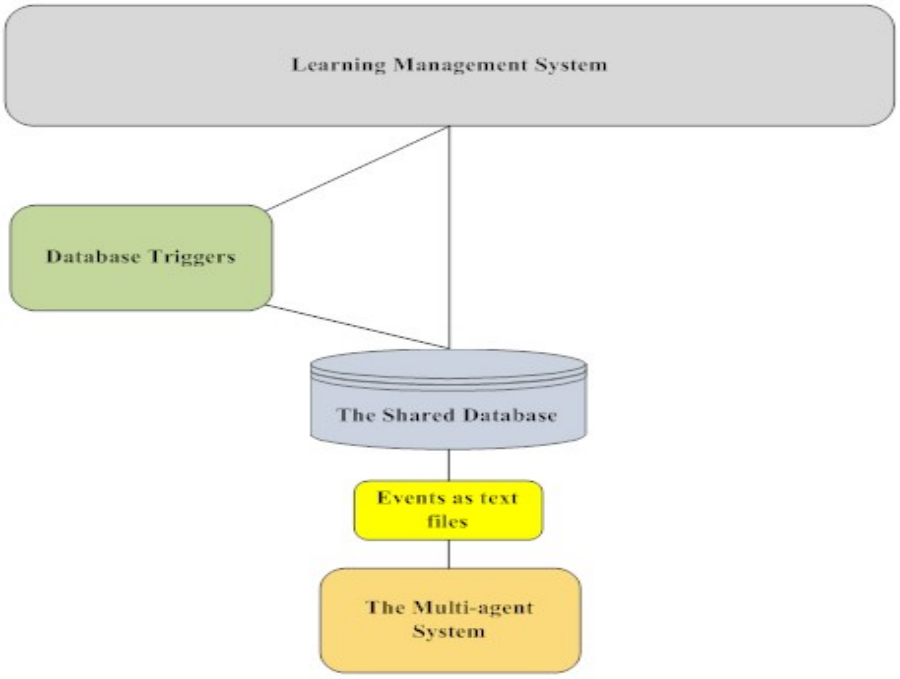


FIGURE 5.5: The prototype structure

The learning management system can be any LMS in which the required database triggers are integrated. Table 5.3 shows each component in the prototype structure with its corresponding component in the proposed architecture, which has been discussed in Section 4.3. The source code of the system prototype is included in Appendix B; it could be used as a reference for researchers who might be interested in the development of agents in the context of e-learning.

TABLE 5.3: Prototype Components Vs Architecture Components

Prototype component	**Architecture component**
Database Triggers	The ECA module
The Shared Database	Shared Database
Events Detection Mechanism	Detection Mechanism
The Multi-agent System	The Multi-agent Module

5.5 Summary

In this chapter, the system prototype of the proposed architecture has been described. The prototype reflects the proposed hybrid approach using the concept of the ECA model and intelligent agents. The hybrid approach was developed to support dynamic real-time adaptivity in e-learning environments based on learners' learning styles. The prototype can be integrated into any LMS since it depends mainly on database triggers, which can be incorporated into the database of the LMS. Beside this, it includes the multi-agent system, which is completely isolated from the design of the LMS. In the following chapter, a real-life problem is demonstrated in which our proposed approach is evaluated using a case study in Moodle.

Chapter 6

Evaluation: A Case Study

Objectives:

- To evaluate the implementation of the proposed hybrid approach
- To develop a validation tool for presenting the results of adaptivity
- To state the significance of our approach among other related work in Virtual Learning Environments

---

6.1 Introduction

This chapter aims to evaluate and validate the implementation of the proposed hybrid approach. In order to achieve this, the evaluation process is based on the assumption that an online course is designed and offered for distance learners using Moodle as the online learning platform. Thus, the proposed approach is integrated and evaluated in Moodle as a case study. Moreover, a validation tool is developed to observe the results of the adaptivity support in detail. The case study presents different live scenarios to show how the proposed approach supports real-time adaptivity for learners based on their preferred learning styles. It also illustrates the interaction of agents to provide learners with adaptive content. Finally, the significance of our approach is stated taking into consideration previous related work, which has been discussed earlier in this thesis.

6.2 A Case Study in Moodle

Moodle is a free online LMS under the terms of the General Public Licence (GPL). It is one of the most popular LMSs and is used by many educational institutions worldwide. Therefore, Moodle was chosen as the Learning Management System in which to validate and integrate our approach. Let us suppose that an online course, namely Computing Essentials, is offered in a particular educational institution, a university for instance. The course is an introduction to data, information and computing. The university offers this online course using Moodle as the LMS so that learners can access the course material. The proposed approach is integrated in the LMS in order to support adaptivity as discussed previously in Chapter 5. The course structure and content are discussed in the following section.

6.2.1 Course Structure and Content

The course structure follows the general adaptive course structure as identified earlier in Section 4.3.1.4. It consists of two main sections: the adaptive and non-adaptive sections. The former is designed to present the actual adaptive content that is dedicated to each learner. The latter is designed to include any content that should be presented to all learners. Let us suppose that all of the course content is created and added to the LMS as shown in Figure 6.1. At this stage, it is very important to mention that this content should be created and designed by the course editors, who should be aware of learning styles and their supportive content (i.e. the suitable content for each learning style). The design of the course content is beyond the scope of this thesis.

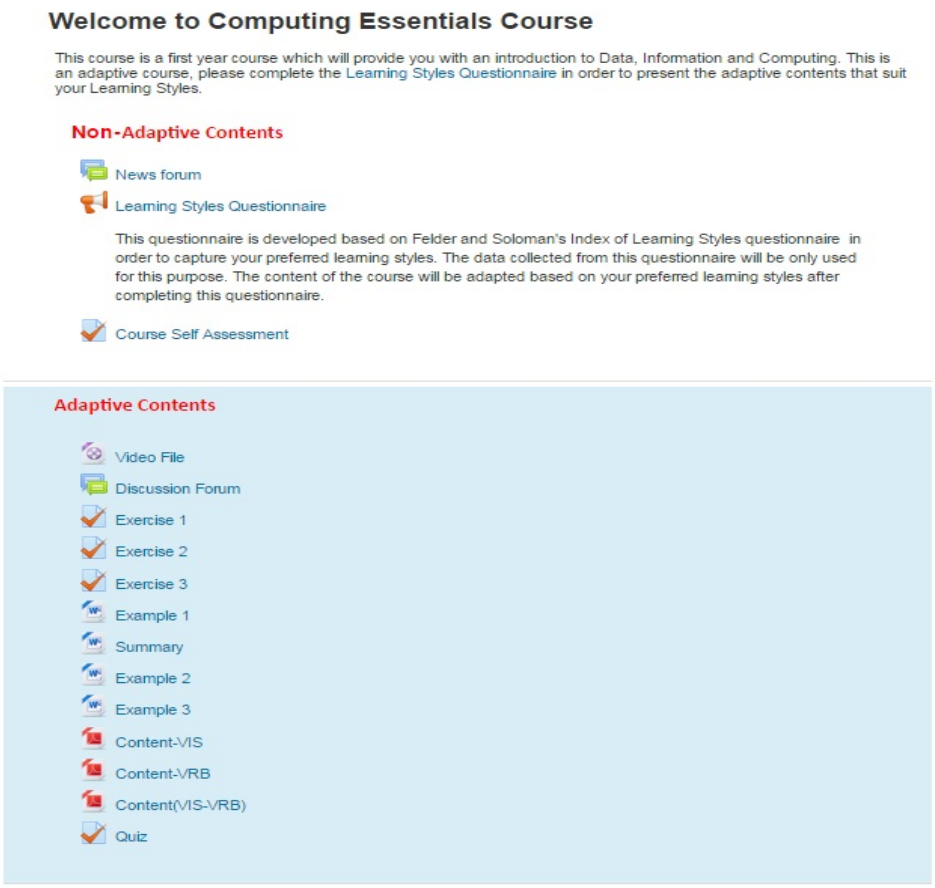


FIGURE 6.1: The course page with all content in Moodle

The LMS deals with the course content as items stored in its database. Each item is associated with an ID and classified into a specific category. For implementation purposes, the course content categories are identified as shown in Table 6.1.

TABLE 6.1: The categories of learning activities and resources

Category ID	**Category Name**
1	Forums
2	Examples
3	Content-Visual
4	Exercises
5	Quizzes
6	Summaries
7	Content-Verbal
8	Content-VISVRB

Each course content item belongs to one of the categories shown in Table 6.1 above. These categories are used in order to differentiate between learning activities and learning resources that are presented to the learners. For example, categories 2,3,6,7 and 8 are considered learning resources whereas categories 1, 4 and 5 are considered learning activities. Moreover, categories 3, 7 and 8 are categories that can hold content that is suitable for visual, verbal or visual-verbal learners respectively. These categories have been defined for the evaluation of the proposed approach. However, the course designers can define any new categories as required. Table 6.2 shows the course content with the IDs and categories in the LMS. The proposed approach deals with the IDs of the content for each course designed in the LMS. Therefore, the LMS is used to create and present the course content while the proposed approach is responsible for adapting this content to a particular learner.

TABLE 6.2: The course content in Moodle

ID in Moodle	**Category**	**Description**
17	4	Exercise 1
24	4	Exercise 2
25	4	Exercise 3
31	5	Quiz
14	1	Discussion Forum
21	2	Example 1
22	2	Example 2
23	2	Example 3
28	3	Video File
29	7	Content-VRB
2	3	Content-VIS
30	8	Content(VIS-VRB)
33	6	Summary

Table 6.2 above shows the course content created in the LMS. This content is created for evaluation purposes and based on the proposed generic course structure. However, the course designers can include any additional content that may support learners. For example, the course designer can define a new category, namely extra material, which consists of supplementary material to support particular learners. The proposed approach can process any course content that can be defined in the LMS. In the following section, different live scenarios are presented to show how our approach provides adaptive content to each learner.

6.3 Evaluation Scenarios

The hybrid approach proposed in this thesis should support real-time adaptivity based on learners' learning styles in any given LMS. Different scenarios are presented to show how adaptivity support is achieved using our approach. As a part of our evaluation, a validation tool is developed in order to simulate the interactions of the agents of the multi-agent system and provide the results of each interaction until the final goal is reached, which is to present adaptive content to learners.

It is probably impossible to mention all of the scenarios that may occur in this case study; there is a high probability of getting different strengths for each learning style depending on the learner's response to the learning styles questionnaire. However, specific scenarios are presented to provide evidence of the applicability of our approach in terms of liveness and safety, and its capability to provide dynamic adaptivity in a timely manner. Moreover, these evaluation scenarios are presented to show that the proposed approach can follow pre-defined pedagogical rules that control the adaptation process; the proposed approach provides dynamism so that course designers can define and modify these rules in order to achieve different adaptation experiences. The scenarios presented in this section provide the full capability of the proposed approach.

6.3.1 Adaptation Process

The adaptation process proposed in this thesis is based on three main adaptive features: the type, number and order of the presented content. The course editors must determine whether or not a category supports a specific learning style as stated earlier in this chapter. This process is crucial for the adaptation process. Let us suppose that the supportive categories for each learning style

are determined by the course designer and stored in the database, the Shared database, as shown in Table 6.3.

TABLE 6.3: Content categories and learning styles

Learning Style	Category	Category Support
ACT	Forums	1
ACT	Quiz	1
ACT	Exercises	1
ACT	Examples	-1
REF	Forums	-1
REF	Quiz	-1
REF	Examples	1
REF	Exercises	-1
SEN	Exercises	1
SEN	Examples	1
SEN	Quiz	1
INT	Quiz	1
INT	Examples	-1
INT	Exercises	-1
VIS	Content(VIS-VRB)	1
VIS	Content-VIS	1
VRB	Content-VRB	1
SEQ	Summary	1
GLO	Summary	1

The abbreviations *ACT*, *REF*, *SEN*, *INT*, *VIS*, *VRB*, *SEQ* and *GLO*, shown in Table 6.3, refer to the learning styles *Activist*, *Reflective*, *Sensitive*, *Intuitive*, *Visual*, *Verbal*, *Sequential* and *Global* respectively. As can be seen from Table 6.3, the column "Category Support" has two values: (1) and (-1), which indicate whether or not the category supports the learning style, respectively. In addition, these values are used in order to determine the recommended amount of content for each category to be presented for each learner. They are used to deal with conflicts between learning styles. For example, a conflict may occur when a particular learning style suggests increasing the number of exercises while another learning style suggests the opposite. In this case, the learner is provided with a moderate number of exercises. It is worth mentioning here

that the content categories, which have no effect on a particular learning style, do not need to be defined. For example, the category "Forums" is shown only against the ACT/REF learning style category as forums have no effect on the other learning styles, as can be seen in the table above.

In the following scenarios, we present the adaptation process for different learners with different learning styles taking into consideration the adaptive features discussed in this thesis. The objective of presenting these scenarios is to show how dynamic adaptivity is achieved in a timely manner using our proposed hybrid approach. Also, these scenarios show how our approach can follow the requirements of adaptation (i.e. pre-defined rules) in order to present adaptive content based on learners' learning styles.

-Scenario 1:

Let us suppose that a learner, "Learner 1", has logged on to the Adaptive Learning Management System (ALMS) and accessed the course for the first time. The course page presented to that learner is shown in Figure 6.2.

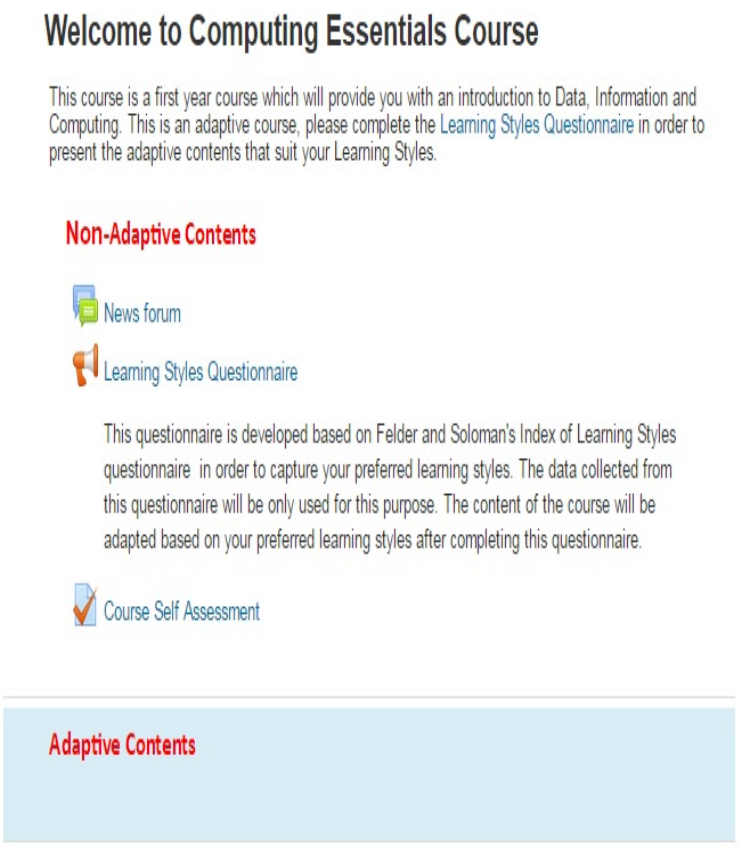


FIGURE 6.2: The course page presented for the first time

As shown in Figure 6.2, the course page is presented without an adaptation effect, before the learning styles questionnaire is completed. Let us suppose that the learner completes the questionnaire and the results are as shown in Table 6.4.

TABLE 6.4: Learning styles questionnaire results for "Learner 1"

Learning Style	**Strength**
ACT	Strong
SEN	Strong
VIS	Strong
SEQ	Strong

Once the learner has completed the learning styles questionnaire, the multi-agent system is informed about this transaction by the database triggers (see Section 5.2.2). Then, the adaptation process takes place via the multi-agent system. Figure 6.3 shows the interactions between agents using the simulation tool namely, the Sniffer, in JADE.

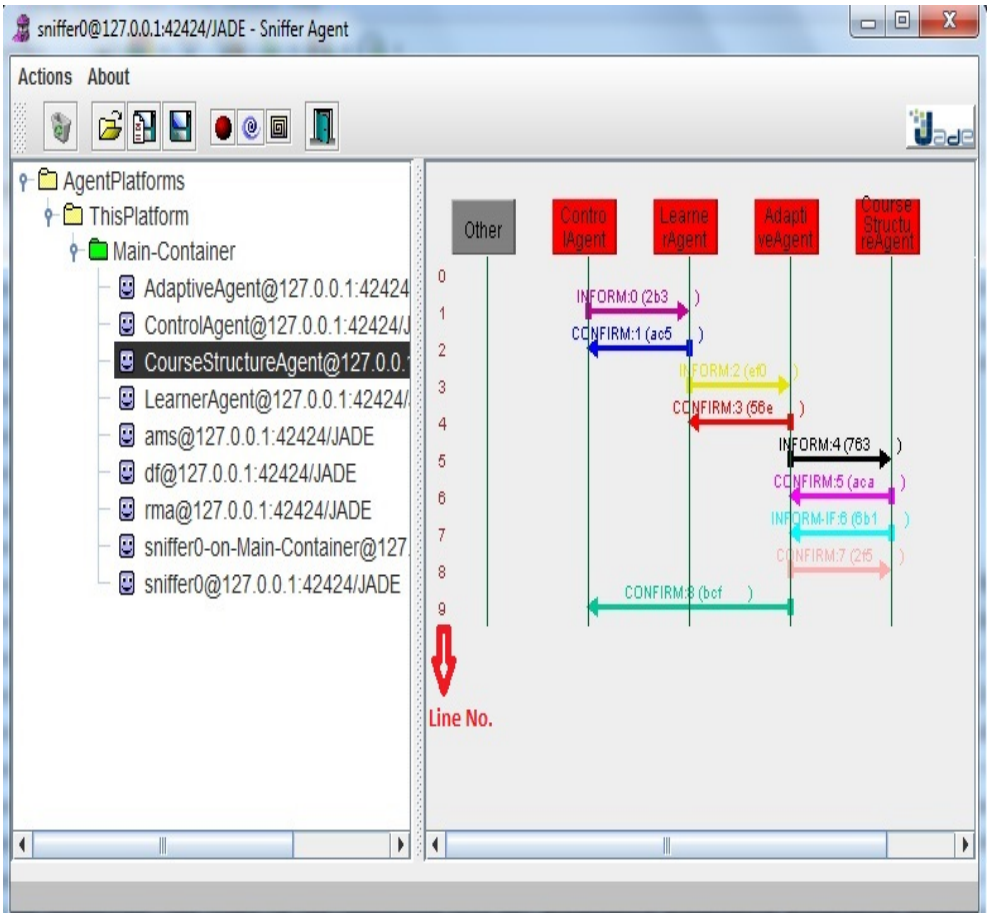


FIGURE 6.3: The interactions of agents

As shown in Figure 6.3 above, the interaction begins from line "1" when the event of completing the questionnaire is received by the control agent from the database. Then, the Control Agent informs the Learner Agent about this event in order to calculate the strength of the learner's learning styles, as discussed in Chapter 4. The preferred learning styles of the learner are processed and stored

in the "Shared" database. The Learner Agent informs the Adaptive Agent that the results of the learning styles are ready for processing. Based on the results, the Adaptive Agent collects suitable content that supports the learner's learning styles from the Course Structure Agent. Finally, the Adaptive Agent applies the adaptive features following any pre-defined pedagogical rules and returns the adaptive content back to the ALMS for the learner, as shown in Figure 6.3 at line "9". Figure 6.4 demonstrates the results of each interaction in detail using the validation tool developed for this purpose.

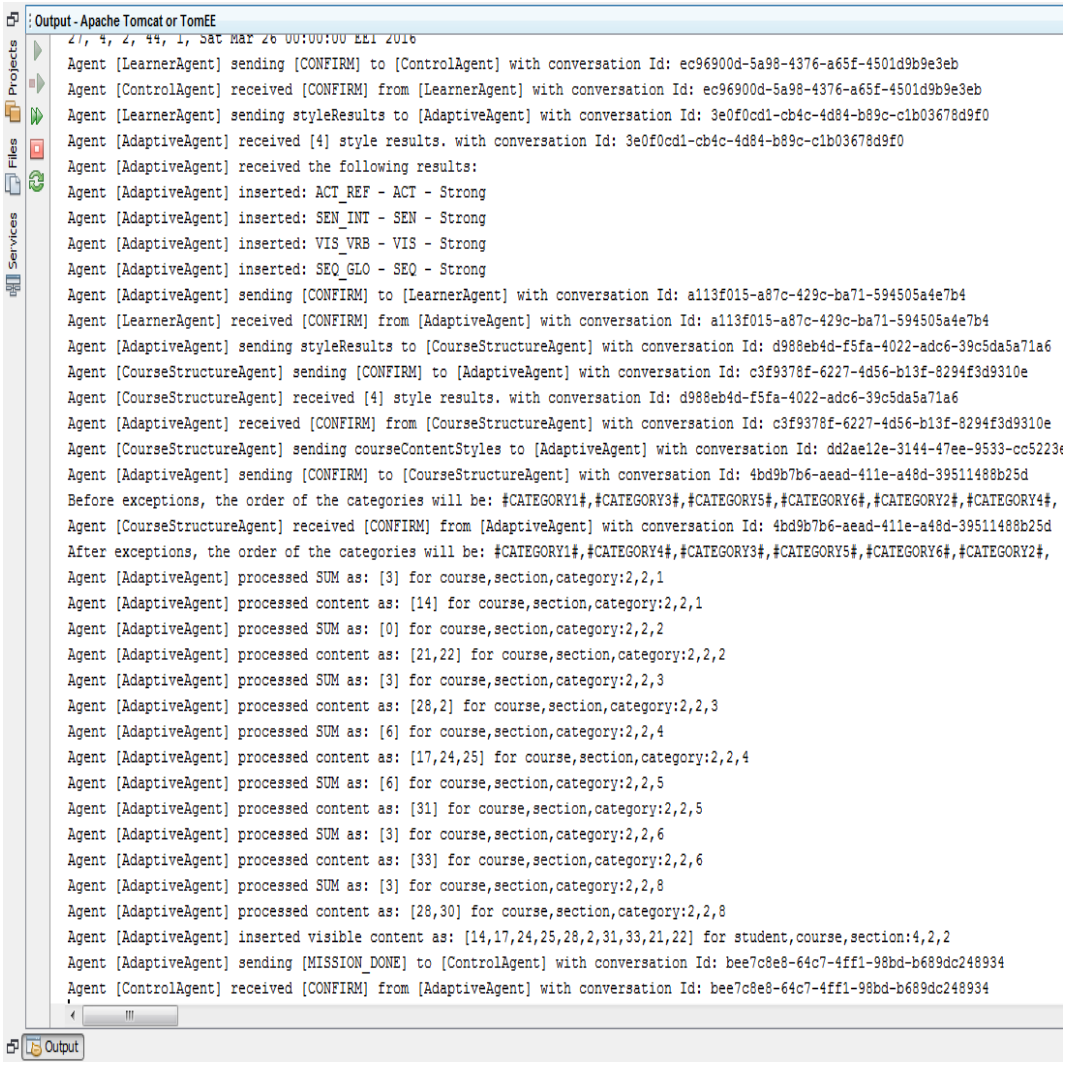


FIGURE 6.4: The results of agents' interaction for "Learner 1"

Figure 6.4 shows the processes of each agent and the interaction with other agents in order to provide "Learner 1" with adaptive content based on the results of the learning styles questionnaire. The first process is initialised when the learner completes the questionnaire. This event is recognised by the database trigger, which is responsible for differentiating between the learning styles questionnaire and other questionnaires defined in the course page. For example, a course page may contain other types of questionnaires that have no effect on the adaptation process.

The Control Agent (CA) is informed about the event using the approach developed and explained in Section 5.3.2. The event is stored as a text file in a specific folder of events. This folder is monitored by the multi-agent system. The CA tells the Learner Agent (LA) to calculate the results of the learning styles questionnaire completed by the learner. The LA calculates the strength of the learning styles based on the Felder and Soloman Index of Learning Styles, as explained in Section 4.3.1.3. The results are sent to the Adaptive Agent in order to communicate with the Course Agent and retrieve the course content that supports the learner's learning styles. Moreover, the Adaptive Agent applies the pre-defined rules that control and determine the number and order of the presented content (i.e. adaptive features). The pre-defined rules are identified by the course designers to provide dynamic adaptivity.

For this scenario and based on the strengths of the learner's learning styles, "Learner 1" is provided with visual content. Also, the number of exercises is increased while the number of examples is reduced for this learner. As stated previously in this section, the adaptive system can deal with conflicts between learning styles. For example, a particular learning style may recommend increasing the number of examples while another may recommend decreasing the number of examples. As a result, the system provides the learner with a

moderate number of examples. This depends on the total number of activities (i.e. number of exercises) defined by the course designer and the strength of the learning styles. For example, if the total number of exercises defined in the LMS were 6, the moderate number of exercises presented would be 3. In this scenario, there is a conflict in the learner's preferred learning styles; as an activist with a strong preference, it is recommended to reduce the number of examples, and as a sensitive learner with a strong preference, it is recommended to increase the number of examples. In this case, the system recommends a moderate number of examples to be presented to this learner. The exercises are presented before the visual content whereas the examples are shown after the visual content. Moreover, a discussion forum is presented to the learner. The summary is presented after the visual content and before the examples. It is very important to mention that the adaptive features, including the type, number and order of the content, are based on the recommendations of the Felder-Silverman learning styles model, as explained previously in Section 4.3.1.5. However, the proposed ALMS supports dynamism so that course designers can modify the adaptive features (i.e. the type, number and order of the content) to achieve a different adaptation experience. In addition, learners can show all of the content at any time without any adaptivity support.

As stated earlier in this chapter, the adaptive process takes place in a timely manner when the learner has completed the learning styles questionnaire. The final result of the presented adaptive content for "Learner 1" is depicted in Figure 6.5.

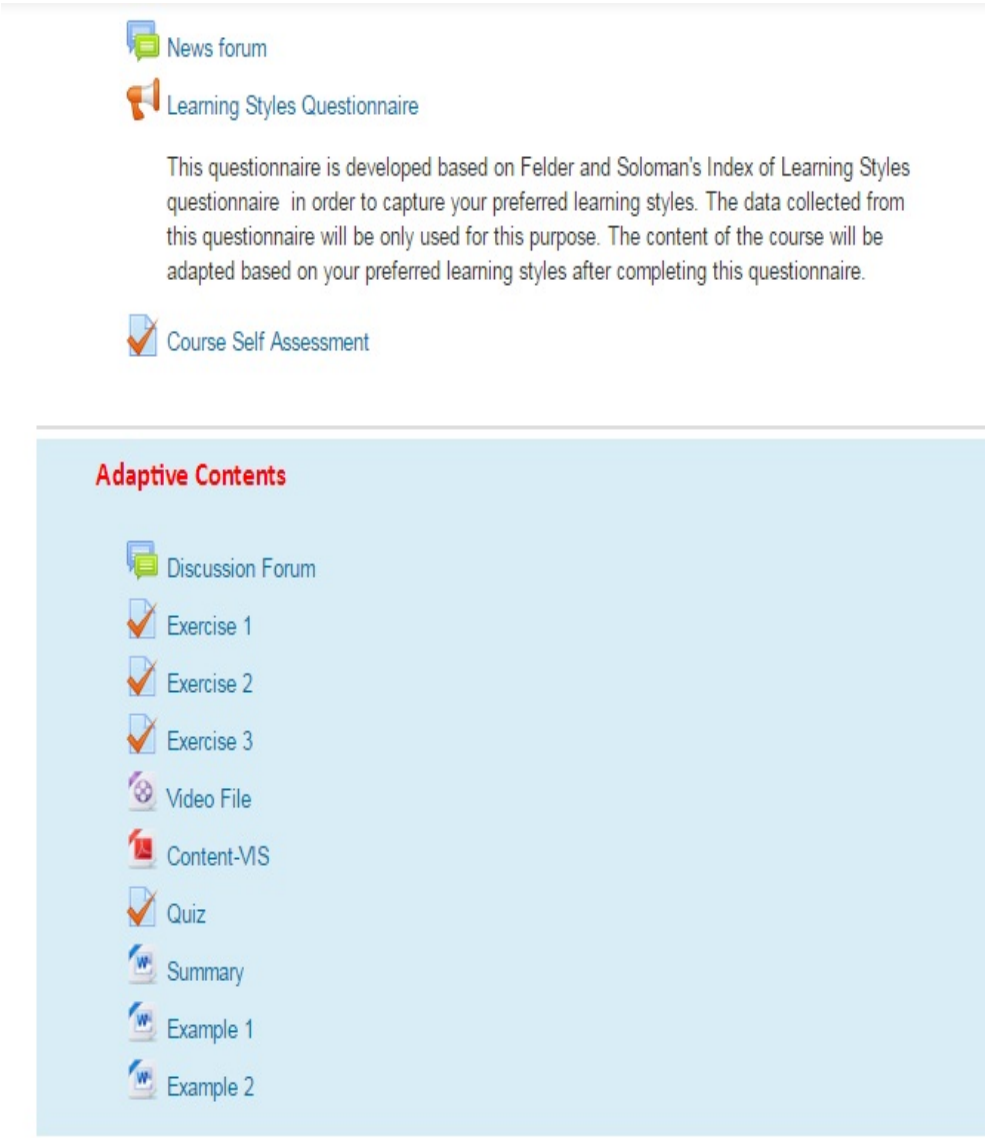


FIGURE 6.5: The adaptive content presented to "Learner 1"

The proposed approach has also been designed to provide feedback to learners on their progress and performance in the course, as mentioned earlier in Chapter 4. When the learner completes the course self-assessment activity (see Figure 6.5 above), the system sends a notification (i.e. an email) to the learner accordingly. The course self-assessment is used to evaluate learners' general performance in the course. The course designers and instructors can

specify the type of feedback to be sent for each learner based on the score received. For example, if the learner's score is less than 6 out of 10, then the system notifies the learner by email about how to improve his/her performance. Let us suppose that "Learner 1" has completed the course self-assessment and the score of 5 is received, which is lower than the required score. Then, the ALMS notifies the learner about his/her progress, as depicted in Figure 6.6.

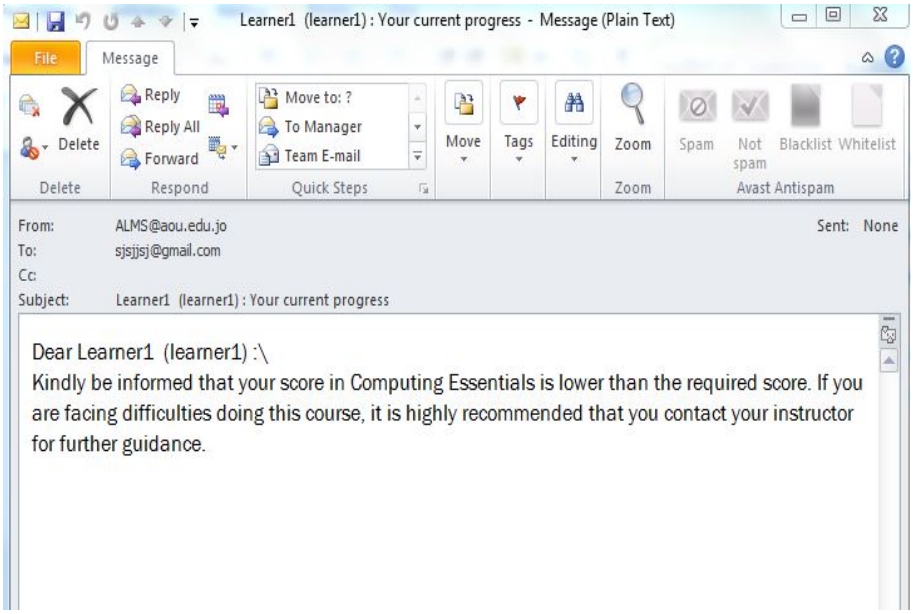


FIGURE 6.6: Feedback to "Learner 1"

-Scenario 2:

This scenario is based on the assumption that another learner, "Learner 2", has accessed the course and completed the learning styles questionnaire. As a result, the multi-agent system responds to this event in a timely manner and calculates the strength of his/her learning styles. The results of the questionnaire are shown in Table 6.5.

TABLE 6.5: Learning styles questionnaire results for "Learner 2"

Learning Style	**Strength**
REF	Strong
INT	Strong
VRB	Strong
GLO	Strong

The course page presented to this learner for the first time is the same as previously depicted in Figure 6.2. Based on the results shown in Table 6.5 above, the learner is provided with adaptive content that suits his/her preferred learning styles by the multi-agent system. The interaction of the agents for this scenario is depicted in Figure 6.7.

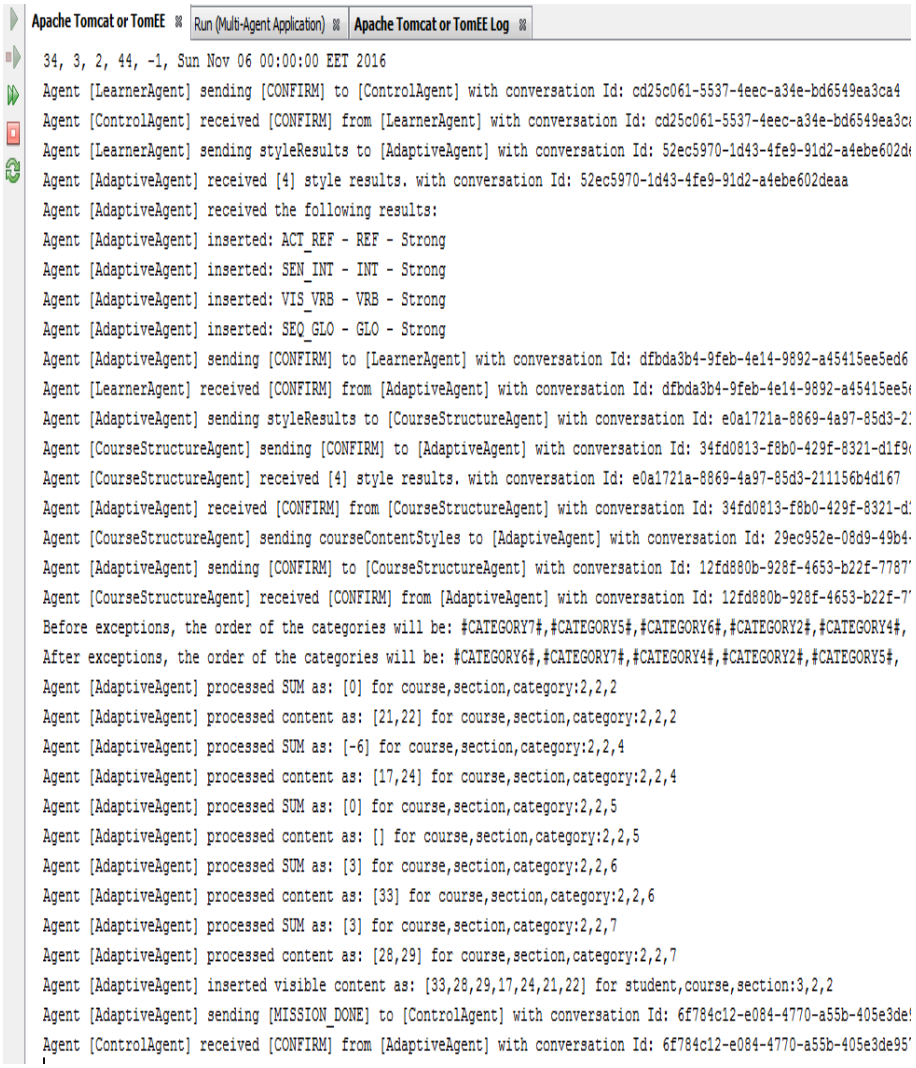


FIGURE 6.7: The results of agents' interaction for "Learner 2"

As can be seen from Table 6.5 and Figure 6.7, the learner is classified based on his/her preferred learning styles as REF, INT, VRB and GLO. The strength of each learning style is Strong. The content dedicated to Verbal learners is presented to this learner instead of Visual content, for instance. Also, as a Reflective learner, the discussion forum is hidden from the adaptive content, and the number of exercises is reduced and shown after the course verbal content. For an Intuitive learning style, the learner is provided with a moderate number of examples, which are shown after the course content. The summary is

presented before the course content to support his/her Global learning style. The adaptive content presented for "Learner 2" is illustrated in Figure 6.8.

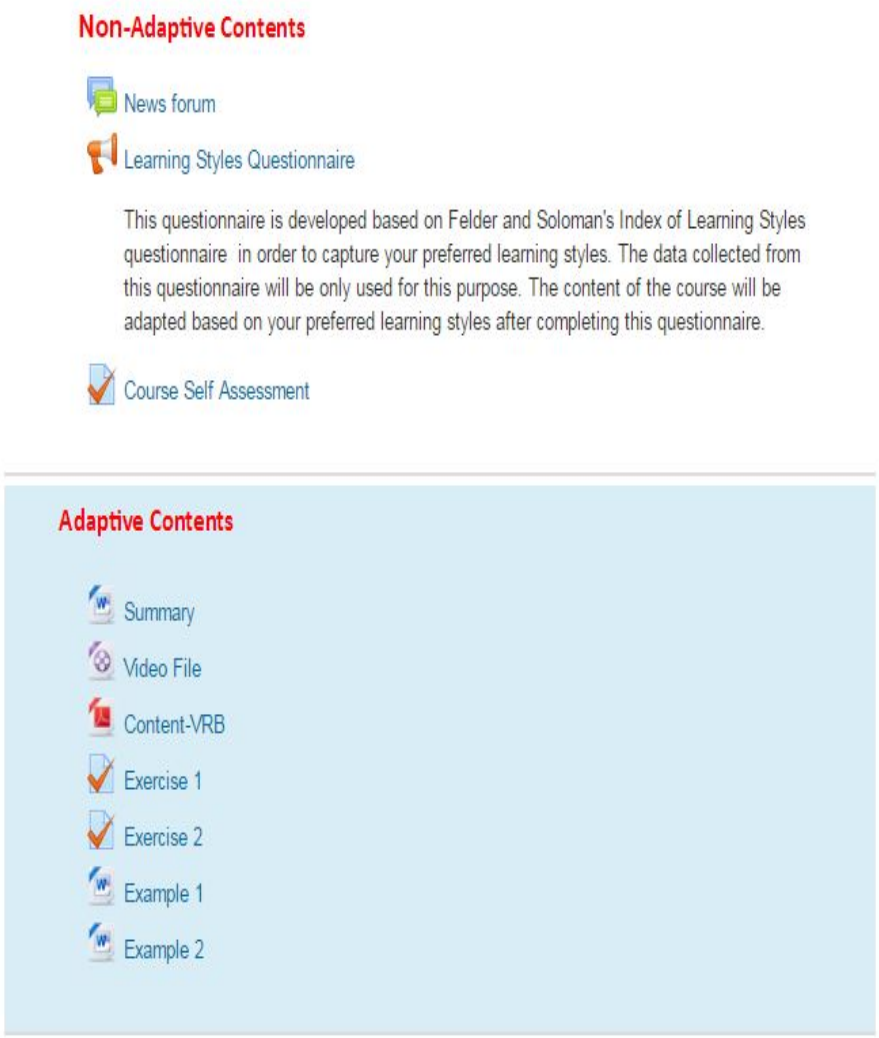


FIGURE 6.8: The adaptive content presented to "Learner 2"

As can be seen from the figure above, the adaptive content presented to "Learner 2" is different from that presented to "Learner 1" (see Figure 6.5). This is due to the differences between their preferred learning styles. For example, "Learner 1", as a visual learner, has been provided with visual content (i.e. Content-VIS) whereas "Learner 2", as a verbal learner, has been provided with verbal content (i.e. Content-VRB). Moreover, no discussion forums have

been presented to "Learner 2" as forums are not supportive for a reflective learner. The number of exercises has been reduced for "Learner 2", as a reflective learner, and these exercises have been presented after the course material (i.e. Content-VRB). In contrast, the number of exercises has been increased for "Learner 1", as an active learner, and these exercises have been presented before the course material (i.e. Content-VIS). As a global learner, "Learner 2" has been provided with a summary, which is shown before the course material.

-Scenario 3:

In this scenario, different strengths of learning styles are demonstrated. Let us suppose that a learner, "Learner 3", has completed the learning styles questionnaire and the results are as shown in Table 6.6.

TABLE 6.6: Learning styles questionnaire results for "Learner 3"

Learning Style	**Strength**
ACT	Balanced
SEN	Balanced
VIS	Balanced
GLO	Moderate

"Learner 3" is classified based on the results of the learning styles questionnaire. The learner has Balanced strengths for ACT, SEN and VIS and Moderate strength for GLO. A balanced strength means that the learner has the same strength in regard to two learning styles in the same category. For example, "Learner 3" has a balanced preference between Activist and Reflective. The interaction of agents in order to generate the suitable adaptive content for "Learner 3" is shown in Figure 6.9.

FIGURE 6.9: The results of agents' interaction for "Learner 3"

The purpose of this scenario is to present how the multi-agent system can deal with different strengths between a learner's learning styles. Based on the learner's learning styles shown in Table 6.6, the learner is provided with visual-verbal content to meet his/her balanced preference in the VIS/VRB learning style category. The number of examples and exercises is reduced for this learner. The summary is shown before the learning material. To recall, the system can decide the number of activities to be presented depending on the strength of the learning styles, the total number of activities defined and the

existence of conflicts between learning styles. The adaptive content presented to "Learner 3" is depicted in Figure 6.10.

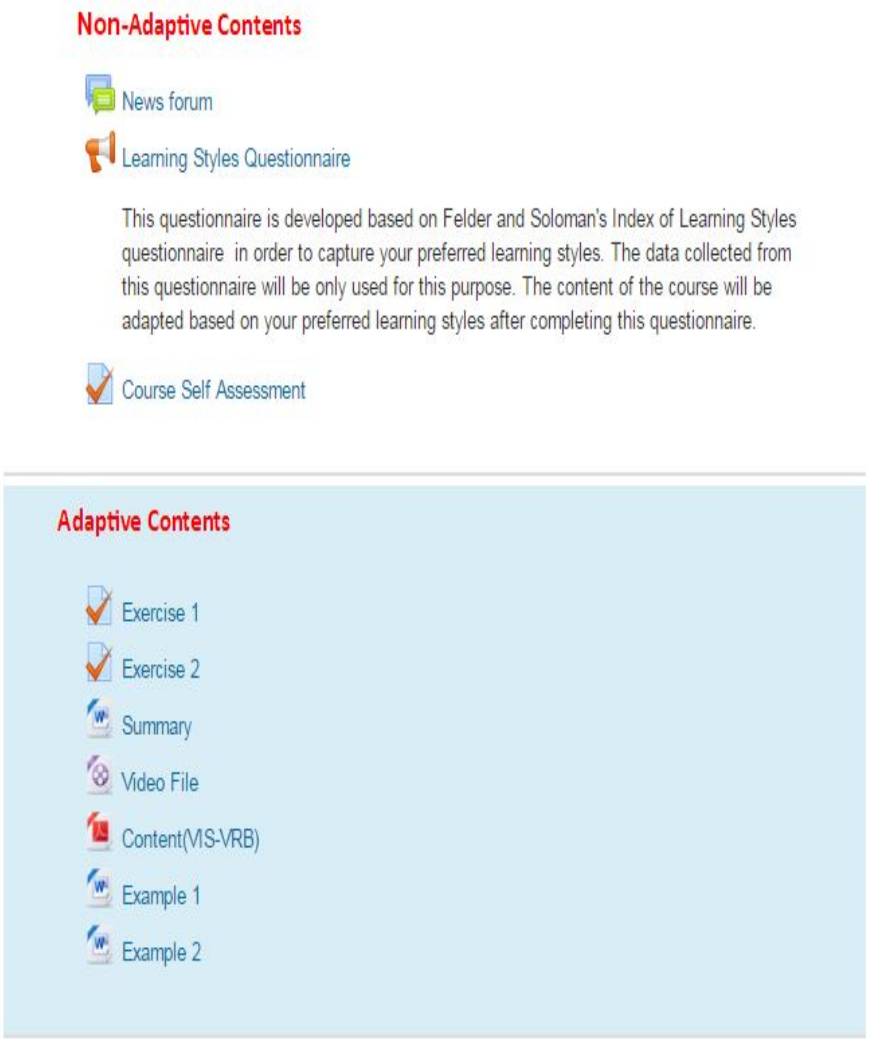


FIGURE 6.10: The adaptive content presented to "Learner 3"

6.4 Discussion

The proposed hybrid approach is validated using different live scenarios in Moodle as a case study. As stated before, it is impossible to mention all of the

cases that might occur with learning styles in the LMS. The objective of demonstrating the aforementioned scenarios is to provide evidence of the applicability of our proposed hybrid approach in a real life problem and how it supports dynamic adaptivity in a timely manner in any Virtual Learning Environment. The scenarios have been chosen to validate our approach in terms of the liveness and safety properties associated with agents' structure. To recall what has been previously stated in this chapter, the proposed approach has been designed to be integrated in any LMS in order to support real time adaptivity. In addition, it supports dynamic adaptivity, which can be controlled based on the pedagogical rule identified by the course designers and instructors. In this thesis, adaptivity support is based on the proposed adaptive features, which are based on the recommendations of the FSLSM. However, our approach can deal with any other adaptive features that might be suitable for each type of learner. For example, the instructor or the course designer can change the order, type and amount of content that might be supportive for learners. It is also important to mention that the aim of this thesis is to propose a novel hybrid approach using the concept of the ECA model and intelligent agents to support adaptivity in any Virtual Learning Environment from a technological perspective. The design of course materials and their association with learning styles are beyond the scope of this thesis.

As part of our evaluation, a validation tool is developed to show how the system behaves to support adaptivity in a timely manner. The run-time results, including the interaction of agents, have been illustrated in the previous scenarios to demonstrate the concept of our proposed approach. Our approach should provide real-time dynamic adaptivity in any LMS, taking into consideration the predefined pedagogical rules, which are defined by course designers. The pedagogical rules that have been applied in our evaluation scenarios can

be summarised as follows:

- If the learning activity is identified as the learning styles questionnaire, then recognise the event and inform the multi-agent system.
- If the learner gets a score that is less than a specific value (for example 6 out of 10) when completing the course self-assessment, notify the learner by email.
- Particular content should be presented in a specific order taking into consideration some exceptions.
- Particular content should be hidden for a specific type of learner.
- Particular content should be presented for a specific type of learner.
- The number of learning resources and activities must comply with learners' preferred learning styles, especially when conflicts between them exist.

The rules mentioned above are specified by the course designers using a dedicated user interface, as stated in Chapter 4. The adaptive system follows the rules (i.e. pedagogical rules) and presents suitable content based on them. Our approach can be categorised as a stand-alone system that can be integrated into any LMS after which the required database triggers are installed. In the following section, we state the significance of our approach in VLEs.

6.5 The Significance of the Proposed Approach

As can be concluded from the literature, adaptivity support in e-learning environments can enhance the learning process, especially when learners' learning styles are considered. Several frameworks and systems have been discussed and reviewed in Sections 2.7 and 3.8. The limitations of these systems have

been identified. To the best of our knowledge, no previous work has been done incorporating the concept of the Event-Condition-Action (ECA) model and intelligent agents as a hybrid architecture to support adaptivity in e-learning environments. Moreover, the proposed approach provides dynamic real-time adaptivity that can be integrated into any LMS. From a technological perspective, the approach provides an efficient tool for expanding the capabilities of LMSs in supporting adaptivity. This study can open up further research on using intelligent agents in order to enhance the learning process in virtual learning environments, especially in Massive Open Online Courses (MOOCs). Therefore, the technical aspects of the proposed approach have been included in the Appendices of this thesis for other researchers who might be interested in using intelligent agents in pedagogy.

6.6 Summary

In this chapter, the system prototype of our approach is evaluated using different live scenarios in Moodle as a case study. The evaluation scenarios have been chosen in order to present how dynamic real-time adaptivity is achieved in the LMS where the system prototype is integrated, and how feedback is provided to learners. Moreover, the scenarios illustrate the behaviour of each component of the system prototype in each stage to provide learners with adaptive content based on their preferred learning styles. Finally, we state the significance of our approach in e-learning environments compared with previous work in the field. In the following chapter, we conclude the work presented in this thesis and state our future directions.

Chapter 7

Conclusion and Future Work

7.1 Research Summary

This thesis proposed a hybrid architecture for supporting adaptivity in e-learning environments based on learners' learning styles. Based on the proposed architecture, a novel approach has been developed to provide dynamic real-time adaptivity in any e-learning system. The work undertaken in this thesis can be summarised as follows:

- In Chapter 1, an introduction to the research background was provided. Also, the research methodology and questions were identified.
- Chapter 2, which is the first part of the literature review, introduced e-learning and e-learning systems and pointed out the drawbacks of the current LMSs as e-learning platforms. Learning styles models were introduced and discussed in detail, and the most common learning styles models in the literature were identified. Moreover, several learning systems incorporating learning styles were investigated, pointing out the limitations of these systems.

- Chapter 3 introduced the technologies used in the design of the proposed architecture. The first part introduced agent technology, a taxonomy of agents and the type of environments where agents can live. In addition, the most popular agent development platforms and agent design methods were thoroughly investigated. In the second part of this chapter, the concept of the ECA model was introduced and the significance of using such a model with agents in the e-learning environment was identified. Finally, some adaptive learning systems incorporating agent technology were reviewed.

- Chapter 4 discussed the design of the hybrid architecture to support adaptivity in e-learning environments in detail. Each component of the architecture was designed, pointing out its role in the whole system. The architecture reflects a novel approach to providing dynamic real-time adaptivity based on learners' learning styles. A learning styles questionnaire was adapted and developed based on the Index of Learning Styles questionnaire of Felder and Soloman. Moreover, a proposed adaptation process and the design of an adaptive course structure were explained. Finally, the adaptation process can be controlled using pre-defined pedagogical rules, which can be modified by course designers.

- In chapter 5, the system prototype of the proposed approach was developed. The multi-agent system was developed based on the design of the multi-agent module discussed in Chapter 4. The ECA model concept was implemented as database triggers in the database of the e-learning system. Moreover, a mechanism was proposed and developed in order to provide the multi-agent system with real-time data about events in the e-learning system. The system prototype was integrated in Moodle as the e-learning environment.

- The evaluation of the proposed approach was discussed in Chapter 6. The evaluation process was based on a case study in Moodle. A set of scenarios was used in order to validate the system prototype and to show how dynamic real-time adaptivity can be achieved in any e-learning environment. Therefore, a validation tool was developed in order to trace the interactions between agents and the events in the e-learning system. The evaluation scenarios illustrated how adaptivity can be controlled following the pre-defined pedagogical rules, which can be updated by the course designers and instructors. At the end of this chapter, the significance of the proposed approach was stated, pointing out the benefits of applying such an approach in LMSs compared with previous work investigated in Chapters 2 and 3.

7.2 Restating Original Contributions

In this thesis, a novel approach has been introduced to support dynamic real-time adaptivity based on learners' learning styles in e-learning environments. The approach has been developed based on the design of a hybrid architecture using agent technology and the concept of the ECA model. It can extend the capabilities of e-learning systems to provide learners with adaptive content based on their learning styles. Moreover, it can be integrated into any e-learning environment after which the required database triggers are installed. The approach offers a dynamic adaptation process that can follow pre-defined pedagogical rules identified by course designers and instructors. In the following points, the original contributions are revisited:

- **C1:** A computational model has been designed to describe the units of computation in the proposed approach. It defines the role of these units

and the communication between them. Moreover, the model introduces the use of agent technology and the concept of the ECA model as a hybrid approach in the context of e-learning systems.

- **C2:** A hybrid architecture has been proposed based on the computational model. It consists of the following components: the ECA module, the Shared database and the multi-agent module. The design of each component has been thoroughly explained. In addition, the learning styles capturing process has been identified. The architecture reflects a novel approach to supporting dynamic real-time adaptivity based on learners' learning styles. An adaptation process has been proposed, including the design of an adaptive course structure. The adaptation process incorporates some adaptive features that are based on the recommendations of the FSLSM. Moreover, it can be controlled and modified by course designers and instructors to meet the requirements of the educational institution.

- **C3:** A system prototype of the approach has been developed and integrated into an e-learning environment (i.e. Moodle). The prototype involves the development of a multi-agent system and the required database triggers. It reflects the design of each component of the hybrid architecture. Also, the learning styles questionnaire, which is adapted from the ILS questionnaire, has been implemented in the e-learning environment in order to capture learners' learning styles. A real-time mechanism has been developed so that the multi-agent system can detect the events from the LMS and provide adaptive course content in a timely manner. The system prototype can be considered as a standalone tool that can be integrated into any e-learning system.

- **C4:** The evaluation of the proposed approach has been discussed. A case study in Moodle has been used following a set of scenarios to show how

adaptivity is achieved by integrating the proposed approach in Moodle. A validation tool has been developed to trace each step in the adaptation process and to validate our approach. Finally, the significance of the approach has been stated with respect to other related work.

7.3 Revisiting Research Aims

In order to answer the research questions mentioned in Chapter 1, the proposed approach has been designed, implemented and evaluated. The broad aim of this research is to extend the capabilities of LMSs to support dynamic real-time adaptivity based on learners' learning styles. The aims of this research are revisited as follows:

Aim (1) To investigate the current state of VLEs, learning styles models and adaptivity in e-learning systems.

In order to meet Aim 1 above, an intensive review has been undertaken of the most popular VLEs and the capabilities and limitations of these environments have been identified. Also, learning styles models have been explained in detail, and one the most popular learning styles models in the literature, the FSLSM, has been identified. Adaptivity in e-learning environments has been explained in addition to several adaptive learning systems that incorporate learning styles. The drawbacks of these adaptive systems have been stated. Aim 1 has been satisfied in Chapter 2.

Aim (2) To introduce the technologies that have been incorporated in the proposed approach.

The concept of intelligent agents has been thoroughly discussed. This includes the types of agents and the environments in which they can operate. Moreover,

several agent development platforms have been reviewed, and one of the common platforms that researchers use in order to build multi-agent systems has been identified. Furthermore, the methodologies used to design and analyse multi-agent systems have been thoroughly discussed in order to choose an appropriate methodology to design the multi-agent module in the proposed architecture. The concept of the ECA model has also been discussed in order to present its role in providing real-time actions, especially in databases. Several proposed architectures and systems that incorporate the concept of agents have been reviewed. None of these systems incorporates the concept of the ECA model and intelligent agents as a hybrid approach. In addition, there is no evidence that they can support dynamic real-time adaptivity in any type of VLE based on learners' learning styles. Aim 2 has been met in Chapter 3.

Aim (3) To propose an architecture that reflects a hybrid approach in order to support dynamic real-time adaptivity in VLEs based on learners' learning styles.

A computational model that describes the behaviour of the main components of the proposed approach has been introduced. Based on this model, an architecture has been developed using the concept of the ECA model and intelligent agents. The components of the architecture have been explained in detail (see Chapter 4). The adaptive features of the proposed approach have also been identified based on the recommendations of the FSLSM. In addition, the design of the learning styles questionnaire and the adaptive course structure has been discussed.

Aim (4) To evaluate the implementation of the proposed approach in a real life problem.

In order to satisfy Aim 4 above, a prototype of the proposed approach has been

developed and integrated in an LMS (i.e. Moodle). Different evaluation scenarios have been chosen to provide evidence of the applicability of the proposed approach. These scenarios show, step-by-step, the adaptation process achieved by our approach. Therefore, a validation tool has been developed to trace the results of adaptation, especially the interaction of agents in order to provide adaptive content to each learner. Aim 4 has been satisfied in Chapters 5 and 6.

7.4 Future Directions

In this thesis, a novel approach has been proposed in order to support dynamic real-time adaptivity in e-learning environments based on learners' learning styles. The proposed approach can be applied in any e-learning environment. However, it could be improved by adding more adaptation experiences; the proposed approach could be extended to support other types of adaptivity such as adaptivity based on learners' knowledge levels and disabilities.

The learning styles capturing process is based on the completion of the learning styles questionnaire. As a future direction, the use of Educational Data Mining techniques could be integrated in the proposed approach to provide better adaptivity experiences in e-learning environments. Educational Data Mining techniques are used to better understand learners in e-learning environments; they are used to build learners' models, which may improve the adaptation process. Also, they can analyse educational data in order to generate specific rules, classifications and predictions to help students in improving their performance.

The proposed approach has been implemented in the open-source e-learning

environment Moodle. Another direction for future work would be to implement and evaluate the approach in more e-learning systems, especially commercial ones. Also, the system prototype of the proposed approach will be implemented in seven branches of Arab Open University in order to evaluate it from the students' and teachers' perspectives.

Bibliography

AHA Project (2016). AHA Project. [online] Available at: http://aha.win.tue.nl/ [Accessed 19 Mar. 2016].

Al-Azawei, A. and Badii, A. (2014). STATE OF THE ART OF LEARNING STYLES-BASED ADAPTIVE EDUCATIONAL HYPERMEDIA SYSTEMS (LS-BAEHSS). *Int. J. Comput. Sci. Inf. Technol*, 6(3):1–19.

Al-Omari, M., Carter, J., and Chiclana, F. (2015). A Proposed Framework to Support Adaptivity in Virtual Learning Environments. In *Proceedings of The European Conference on Technology in the Classroom 2015, Brighton, United Kingdom*, pages 257–264.

Al-Omari, M., Carter, J., and Chiclana, F. (2016). A hybrid approach for supporting adaptivity in e-learning environments. *International Journal of Information and Learning Technology*, 33(5):333–348.

Alexandru, A., Tirziu, E., Tudora, E., and Bica, O. (2015). Enhanced education by using intelligent agents in multi-agent adaptive e-learning systems. *Studies in Informatics and Control*, 24(1):13–22.

Alghamdi, T. M. K. (2013). *Policy-based Runtime Tracking for E-learning Environments*. PhD thesis, De Montfort University.

Ammar, M. B., Neji, M., and Alimi, A. M. (2005). Emotional multiagents system for peer to peer e-learning (EMASPEL). In *Proceedings of the 5th WSEAS International Conference on Distance Learning and Web Engineering table of contents,*

Chicago, pages 164–170.

Anghel, C. and Salomie, I. (2003). JADE Based solutions for knowledge assessment in eLearning Environments. *EXP - in search of innovation (Special Issue on JADE)*.

Austin, J. L. (1962). *How to do things with words*. Harvard University Press.

Bellifemine, F., Caire, G., and Greenwood, D. (2007). *Developing Multi-Agent Systems with JADE*. John Wiley & Sons Ltd, England.

Blackboard.com (2017). Blackboard. [online] Available at: www.blackboard.com [Accessed 19 Jan. 2017].

Bokhari, M. U. and Ahmad, S. (2013). Design for interactive e-learning based upon multi-agent system: I-mbls. In *Confluence 2013: The Next Generation Information Technology Summit (4th International Conference)*, pages 456–460. IET.

Bra, P. D., Aerts, A., Berden, B., de Lange, B., Rousseau, B., Santic, T., Smits, D., Stash, N., De Bra, P., and de Lange, B. (2003). AHA! The adaptive hypermedia architecture. *Proceedings of the fourteenth ACM conference on Hypertext and hypermedia - HYPERTEXT '03*, 4:81.

Bra, P. D. and Calvi, L. (1998). AHA! An open adaptive hypermedia architecture. *New Review of Hypermedia and Multimedia*, 4(1):115–139.

Bresciani, P., Perini, A., Giorgini, P., Giunchiglia, F., and Mylopoulos, J. (2004). Tropos: An agent-oriented software development methodology. *Autonomous Agents and Multi-Agent Systems*, 8(3):203–236.

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

Brusilovsky, P. (2000). Adaptive hypermedia: From intelligent tutoring systems to Web-based education. In *Intelligent Tutoring Systems*, pages 1–7. Springer Berlin Heidelberg.

Brusilovsky, P. (2001). Adaptive Hypermedia. *User Modeling and User-Adapted Interaction*, 11:87–110.

Brusilovsky, P. (2004). KnowledgeTree: A distributed architecture for adaptive e-learning. In *Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters*, pages 104–113. ACM.

Brusilovsky, P., Eklund, J., and Schwarz, E. (1998). Web-based education for all: a tool for development adaptive courseware. *Computer Networks and ISDN Systems*, 30(1):291–300.

Carter, J., Chiclana, F., and Al-Omari, M. (2015). E-Learning for Distance Students: A Case Study from a UK Masters Programme. In *Proceedings of The European Conference on Technology in the Classroom 2015, Brighton, United Kingdom*, pages 233–242.

Carver, C. A., Howard, R. A., and Lane, W. D. (1999). Addressing different learning styles through course hypermedia. *IEEE Transactions on Education*, 42(1):33–38.

Cha, H. J., Kim, Y. S., Park, S. H., Yoon, T. B., Jung, Y. M., and Lee, J.-H. (2006). Learning styles diagnosis based on user interface behaviors for the customization of learning interfaces in an intelligent tutoring system. In *Intelligent tutoring systems*, pages 513–524. Springer.

Chang, Y.-H. and Chen, Y.-Y. (2012). A mashup-based adaptive learning system. In *International Conference on Machine Learning and Cybernetics (ICMLC)*, volume 5, pages 1721–1726, Xian. IEEE.

Clark, R. C. and Mayer, R. E. (2011). *E-learning and the science of instruction: Proven guidelines for consumers and designers of multimedia learning*. John Wiley & Sons.

Cook, D. A. (2007). Web-based learning: pros, cons and controversies. *Clinical Medicine*, 7(1):37–42.

Corbett, A. T. and Anderson, J. R. (2001). Locus of feedback control in computer-based tutoring: Impact on learning rate, achievement and attitudes. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 245–252. ACM.

Corbett, A. T., Koedinger, K. R., and Anderson, J. R. (1997). Intelligent tutoring systems. *Handbook of human-computer interaction*, 5:849–874.

Cosgrave, R., Rísquez, A., Logan-Phelan, T., Farrelly, T., Costello, E., McAvinia, C., Palmer, M., Cooper, R., Harding, N., and Vaughan, N. (2011). Usage and uptake of virtual learning environments and technology assisted learning: Findings from a multi institutional, multi year comparative study. *AISHE-J*, 3(1).

Crnkovic, G. D. (2010). Constructive research and info-computational knowledge generation. In *Model-Based Reasoning in Science and Technology*, pages 359–380. Springer.

Csapo, N. and Hayen, R. (2006). The role of learning styles in the teaching/learning process. *Issues in information systems*, 7(1):129–133.

Denecke, K. (2012). *Event-Driven Surveillance: Possibilities and Challenges*. Springer Science & Business Media.

Dias, S. B., Diniz, J. A., and Hadjileontiadis, L. J. (2014). *E-Learning Exequibility in the Information and Knowledge Society*, pages 3–19. Springer International

Publishing, Cham.

Dunn, R. and Dunn, K. (1974). Learning style as a criterion for placement in alternative programs. *The Phi Delta Kappan*, 56(4):275–278.

Dunn, R. S. and Griggs, S. A. (2007). *Synthesis of the Dunn and Dunn Learning-Style Model Research: Who, what, when, where, and so what?* St. John's University Press.

Dunn, R., Dunn, K., Price, G. E. (1996). Learning Style Inventory.

Ellis, R. A., Ginns, P., and Piggott, L. (2009). E-learning in higher education: some key aspects and their relationship to approaches to study. *Higher Education Research & Development*, 28(3):303–318.

Entwistle, N., Hanley, M., and Hounsell, D. (1979). Identifying distinctive approaches to studying. *Higher education*, 8(4):365–380.

Etzioni, O. and Weld, D. (1994). A softbot-based interface to the internet. *Communications of the ACM*, 37(7):72–76.

Felder, R. and Silverman, L. (1988). Learning and Teaching Styles in Engineering Education. *Engineering Education*, 78(7):674–681.

Felder, R. M. and Spurlin, J. (2005). Applications, reliability and validity of the index of learning styles. *International journal of engineering education*, 21(1):103–112.

Foundation for Intelligent Physical Agents (2016). Major publicly available implementations of agent platforms which conform to the FIPA Specifications. [online] Available at: http://www.fipa.org/resources/livesystems.html [Accessed 19 Mar. 2016].

Fumero, A. (2006). EDUWEB 2.0 - iCamp & N-Gen Educational Web. In *Proceedings of WEBIST conference*, pages 299–304.

Gilbert, J. E. and Han, C. Y. (1999). Arthur: An adaptive instruction system based on learning styles. In *International Conference on Mathematics/Science Education and Technology*, pages 100–105.

Giorgini, P., Mylopoulos, J., and Sebastiani, R. (2005). Goal-oriented requirements analysis and reasoning in the tropos methodology. *Engineering Applications of Artificial Intelligence*, 18(2):159–171.

Graf, S. (2007). Adaptivity in learning management systems focussing on learning styles.

Graven, O. H., Helland, M., and MacKinnon, L. (2006). The influence of staff use of a virtual learning environment on student satisfaction. In *7th International Conference on Information Technology Based Higher Education and Training.*, pages 423–441. IEEE.

Gregorc, A. F. (1982). *An adult's guide to style*. Gregorc Associates Columbia, Conn.

Gros, B. and García-Peñalvo, F. J. (2016). Future trends in the design strategies and technological affordances of e-learning. *Learning, Design, and Technology: An International Compendium of Theory, Research, Practice, and Policy*, pages 1–23.

Hayakawa, T., Higashino, M., Takahashi, K., Kawamura, T., and Sugahara, K. (2012). Management of multimedia data for streaming on a distributed e-learning system. In *26th International Conference on Advanced Information Networking and Applications Workshops (WAINA)*, pages 1282–1285. IEEE.

Honey, P. and Mumford, A. (1982). *The Manual of Learning Styles*. Peter Honey, Maidenhead.

Honey, P. and Mumford, A. (1992). *The manual of learning styles*. Peter Honey Publications, 3 edition.

Huang, E. Y., Lin, S. W., and Huang, T. K. (2012). What type of learning style leads to online participation in the mixed-mode e-learning environment? A study of software usage instruction. *Computers & Education*, 58(1):338–349.

Jennings, N. and Wooldridge, M. J. (1998). *Agent technology: foundations, applications, and markets*. Springer Science & Business Media.

Jennings, N. R., Corera, J. M., and Laresgoiti, I. (1995). Developing Industrial Multi-Agent Systems. In *ICMAS*, pages 423–430.

Jennings, N. R., Sycara, K., and Wooldridge, M. (1998). A roadmap of agent research and development. *Autonomous agents and multi-agent systems*, 1(1):7–38.

Jung, C. (1923). *Psychological Types*. Pantheon Books, London.

Kolb, A. and Kolb, D. A. (2005). The kolb learning style inventory—version 3.1.

Kolb, D. A. (1976). *Learning Style Inventory: Technical Manual*. McBer.

Kolb, D. A. (1984). *Experiential Learning: Experience as the Source of Learning and Development*. Prentice-Hall, Englewood Cliffs, New Jersey.

Kovalan, A. and Balasubramanian, N. (2008). Quality assessment technique (qat) of virtual learning resources (vlr) in teacher education. In *TENCON 2008-2008 IEEE Region 10 Conference*, pages 1–6. IEEE.

Kravari, K. and Bassiliades, N. (2015). A survey of agent platforms. *Journal of Artificial Societies and Social Simulation*, 18(1):11.

Kroenke, D. M. and Auer, D. (2012). *Database Concepts*. Prentice Hall.

Labro, E. and Tuomela, T.-S. (2003). On bringing more action into management accounting research: process considerations based on two constructive case studies. *European Accounting Review*, 12(3):409–442.

Li, F. W., Lau, R. W., and Dharmendran, P. (2009). A three-tier profiling framework for adaptive e-learning. In *International Conference on Web-Based Learning*, pages 235–244. Springer.

Lim, E. H., Wan Ahmad, W. F., and Hashim, A. S. (2017). *Enhancement of Learning Management System by Integrating Learning Styles and Adaptive Courses*, pages 211–218. Springer International Publishing, Cham.

Litzinger, T. A., Lee, S. H., Wise, J. C., and Felder, R. M. (2007). A psychometric study of the index of learning styles©. *Journal of Engineering Education*, 96(4):309–319.

London, M. (2011). *The Oxford handbook of lifelong learning*. Oxford University Press.

Lukka, K. (2003). The constructive research approach. *Case study research in logistics. Publications of the Turku School of Economics and Business Administration, Series B*, 1(2003):83–101.

Maes, P. (1994). Agents that reduce work and information overload. *Communications of the ACM*, 37(7):30–40.

Mallon, D. (2010). Learning Management Systems 2011 (Executive Summary). Technical report, Bersin and Associates.

Martínez-Caro, E., Cegarra-Navarro, J. G., and Cepeda-Carrión, G. (2015). An application of the performance-evaluation model for e-learning quality in higher education. *Total Quality Management & Business Excellence*, 26(5-6):632–647.

McGill, T. J., Klobas, J. E., and Renzi, S. (2014). Critical success factors for the continuation of e-learning initiatives. *The Internet and Higher Education*, 22:24–36.

McLeod, S. (2010). Kolb - Learning Styles. [online] Available at: http://www.simplypsychology.org/learning-kolb.html [Accessed 19 Mar. 2016].

McLoughlin, C. (1999). The implications of the research literature on learning styles for the design of instructional material. *Australasian Journal of Educational Technology*, 15(3).

Merrill, M. D. (1983). Component Display Theory. In Reigeluth, C., editor, *Instructional Design Theories and Models: An Overview of Their Current Status*, pages 279–333. Lawrence Erlbaum Association, Hillsdale, New Jersey.

Mette, K. and Thomas, C. G. (1994). Adaptivity: system-initiated individualization. In Oppermann, R., editor, *Adaptive user support: ergonomic design of manually and automatically adaptable software*, chapter Adaptivity, pages 67–96. Lawrence Erlbaum Associates, Inc., Hillsdale, NJ.

MLA (2017). Inspiring Learning. [online] Available at: http://www.inspiringlearningforall.gov.uk/learning/ [Accessed 14 Jan. 2017].

Moodle.org (2017). Moodle Open source Learning platform. [online] Available at: https://moodle.org/ [Accessed 19 Jan. 2017].

Morales-Rodríguez, M. L., Ramírez-Saldivar, J. A., Sánchez-Solís, J. P., and Hernández-Ramírez, A. (2012). Design of an Intelligent Agent for Personalization of Moodle's Contents. *Research in Computing Science*, 56:11–17.

Musliner, D. J., Durfee, E. H., and Shin, K. G. (1993). CIRCA: A cooperative intelligent real-time control architecture. *Systems, Man and Cybernetics, IEEE Transactions on*, 23(6):1561–1574.

Musumba, G. W., Oboko, R. O., and Nyongesa, H. O. (2013). Agent-based adaptive e-learning model for any learning management system. *International Journal of Machine Learning and Applications*, 2:1–9.

Myers, I. B. (1998). *Introduction to Type*. Mountain View, CA: CPP, Inc.

Myers and Briggs Foundation (2017). C G Jungs Theory. [online] Available at: http://www.myersbriggs.org/my-mbti-personality-type/mbti-basics/c-g-jungs-theory.htm [Accessed 10 Mar. 2017].

Myers-Briggs, I. (1962). The myers-briggs type indicator manual. *Prinecton, NJ: Educational Testing Service*.

Mylonakis, N. and Cullough, C. (2003). Adaptive Hypermedia – its role in lifelong learning.

Nedev, D. and Nedeva, V. (2006). Aspects of multi-agent system application in e-learning. In *International Scientific Conference Computer Science*, pages 1022–1027.

Noaman, A. Y., Ragab, A. H. M., Madbouly, A. I., Khedra, A. M., and Fayoumi, A. G. (2017). Higher education quality assessment model: towards achieving educational quality standard. *Studies in Higher Education*, 42(1):23–46.

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

O'Leary, R. and Ramsden, A. (2002). Virtual learning environments.

Ong, J. and Ramachandran, S. (2000). Intelligent tutoring systems: the what and the how. *Learning Circuits*.

Ong, J. and Ramachandran, S. (2003). Intelligent tutoring systems: Using AI to improve training performance and ROI. Stottler Henke Associates. *Inc. San Mateo CA*.

Oppermann, R. (1994). *Adaptive user support ergonomic design of manually and automatically adaptable software*. CRC Press.

Oracle (2016). CREATE TRIGGER Syntax. [online] Available at: https://dev.mysql.com/doc/refman/5.5/en/create-trigger.html [Accessed 19 Mar. 2016].

Oxford University Press (2017). Learn about Virtual Learning Environment/Course Management System content. [online] Available at: http://global.oup.com/uk/orc/learnvle/ [Accessed 19 Jan. 2017].

Padgham, L. and Winikoff, M. (2002). Prometheus: A pragmatic methodology for engineering intelligent agents. In *Proceedings of the OOPSLA 2002 Workshop on Agent-Oriented Methodologies*, pages 97–108.

Papamarkos, G., Poulovassilis, A., and Wood, P. (2003). Event-Condition-Action Rule Languages for the Semantic Web. *SWDB*.

Papanikolaou, K. A., Grigoriadou, M., Kornilakis, H., and Magoulas, G. D. (2003). Personalizing the Interaction in a Web-based Educational Hypermedia System: the case of INSPIRE. *User modeling and user-adapted interaction*, 13(3):213–267.

Pappas, C. (2015). Get Them Hooked: 6 Tips To Improve Learner Retention In eLearning Courses And Avoid Drop-Outs. [online] Available at: http://elearningindustry.com/6-tips-to-improve-learner-retention-in-elearning-courses-and-avoid-drop-outs [Accessed 19 Mar. 2016].

Paramythis, A. and Loidl-Reisinger, S. (2004). Adaptive learning environments and e-learning standards. *Electronic Journal on e-Learning*, 2(1).

Pask, G. (1976). Styles and strategies of learning. *British journal of educational psychology*, 46(2):128–148.

Phobun, P. and Vicheanpanya, J. (2010). Adaptive intelligent tutoring systems for e-learning systems. *Procedia-Social and Behavioral Sciences*, 2(2):4064–4069.

Pitigala Liyanage, M. P., Gunawardena, K. S. L., and Hirakawa, M. (2013). A framework for adaptive learning management systems using learning styles. In *2013 International Conference on Advances in ICT for Emerging Regions (ICTer)*, pages 261–265, Colombo. IEEE.

Poulovassilis, A., Papamarkos, G., and Wood, P. (2006). Event-condition-action rule languages for the semantic web. *In Current Trends in Database Technology–EDBT*, pages 855–864.

Reigeluth, C. M. and Stein, F. (1983). The Elaboration Theory of Instruction. In Reigeluth, C., editor, *Instructional Design Theories and Models: An Overview of Their Current Status*, pages 335–381. Lawrence Erlbaum Associates, Hillsdale, New Jersey.

Riding, R. J. (1991). Cognitive styles analysis. *Learning and Training Technology, Birmingham*.

Rundle, S. M. and Dunn, R. (2000). The guide to individual excellence: A self directed guide to learning and performance solutions. *New York, Performance Concepts International*.

Russell, S. and Norvig, P. (1995). *Artificial intelligence: a modern approach*. Prentice Hall, Englewood Cliffs, N.J.

Santos, J. M., Anido, L., and Llamas, M. (2003). On the use of e-learning standards in adaptive learning systems. In *Proceedings - 3rd IEEE International Conference on Advanced Learning Technologies, ICALT 2003*, page 480.

Shang, Y., Shi, H., and Chen, S.-S. (2001). An intelligent distributed environment for active learning. *Journal on Educational Resources in Computing (JERIC)*, 1(2es):4.

Shardanand, U. and Maes, P. (1995). Social information filtering: algorithms for automating "word of mouth". In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 210–217. ACM Press/Addison-Wesley Publishing Co.

Soloman, B. A. and Felder, R. M. (2005). Index of learning styles questionnaire. [online] Available at: http://www.engr.ncsu.edu/learningstyles/ilsweb.html [Accessed 19 Mar. 2016].

Stash, N., Cristea, A. I., and De Bra, P. (2006). Adaptation to learning styles in e-learning: Approach evaluation. In *Proceedings of World Conference on E-Learning in Corporate, Government, Healthcare, and Higher Education*, pages 284–291. AACE.

The University of Texas at Austin (2017). ALPs Assessment and Evaluation Tools. [online] Available at: http://www.me.utexas.edu/ alps/alpAssessment/ [Accessed 14 Feb. 2017].

Titmuss, R., Crabtree, I. B., and Winter, C. S. (1997). Agents, mobility and multimedia information. In *Software Agents and Soft Computing Towards Enhancing Machine Intelligence*, pages 146–159. Springer.

Tomic, D., Muftic, O., and Biocic, B. (2011). A novel scheduling approach of

e-learning content on cloud computing infrastructure. In *MIPRO, 2011 Proceedings of the 34th International Convention*, pages 1443–1446. IEEE.

Trikha, N. and Godbole, A. (2016). Adaptive e-learning system using hybrid approach. In *2016 International Conference on Inventive Computation Technologies (ICICT)*, volume 2, pages 1–4. IEEE.

Tryllian (2016). Agent Dev Kit. [online] Available at: http://www.tryllian.com/adk.html [Accessed 19 Mar. 2016].

University of Leicester (2017). Learning Theories. [online] Available at: http://www2.le.ac.uk/departments/gradschool/training/eresources/teaching/theories/theories [Accessed 15 Feb. 2017].

Viccari, R. M., Ovalle, D. A., and Jiménez, J. A. (2007). ALLEGRO: Teaching/Learning Multi-Agent Environment using Instructional Planning and Cases-Based Reasoning (CBR). *CLEI Electronic Journal*, 10(1).

Villaverde, J., Godoy, D., and Amandi, A. (2006). Learning styles' recognition in e-learning environments with feed-forward neural networks. *Journal of Computer Assisted Learning*, 22(3):197–206.

Viola, S. R., Graf, S., and Leo, T. (2006). Analysis of Felder-Silverman index of learning styles by a data-driven statistical approach. In *Multimedia, 2006. ISM'06. Eighth IEEE International Symposium on*, pages 959–964. IEEE.

Virvou, M. and Kabassi, K. (2002). F-SMILE: An intelligent multi-agent learning environment. In *Proceedings of International Conference on Advanced Learning Technologies-ICALT*. Citeseer.

Walker, R., Voce, J., Nicholls, J., Swift, E., and Ahmed, J. (2014). 2014 Survey of technology enhanced learning for higher education in the UK. Technical

report, Oxford: Universities and Colleges Information Systems Association (UCISA).

Wavish, P. (1996). Situated action approach to implementing characters in computer games. *Applied Artificial Intelligence*, 10(1):53–74.

Webber, C., Bergia, L., Pesty, S., and Balacheff, N. (2001). The Baghera project: a multi-agent architecture for human learning. In *Workshop-Multi-Agent Architectures for Distributed Learning Environments. In Proceedings International Conference on AI and Education. San Antonio, Texas*.

Weiss, G., editor (2013). *Multiagent Systems*. The MIT Press, Cambridge, MA, USA, 2nd edition.

Wolf, C. (2003). iWeaver: towards' learning style'-based e-learning in computer science education. In *Proceedings of the fifth Australasian conference on Computing education-Volume 20*, pages 273–279. Australian Computer Society, Inc.

Wooldridge, M. (2009). *An introduction to multiagent systems*. John Wiley & Sons, 2 edition.

Wooldridge, M. and Jennings, N. R. (1995). Intelligent agents: Theory and practice. *The knowledge engineering review*, 10(02):115–152.

Wooldridge, M., Jennings, N. R., and Kinny, D. (2000). The Gaia methodology for agent-oriented analysis and design. *Autonomous Agents and multi-agent systems*, 3(3):285–312.

Xu, D., Huang, W. W., Wang, H., and Heales, J. (2014). Enhancing e-learning effectiveness using an intelligent agent-supported personalized virtual learning environment: An empirical investigation. *Information & Management*, 51(4):430–440.

Yang, T.-C., Hwang, G.-J., and Yang, S. J.-H. (2013). Development of an Adaptive Learning System with Multiple Perspectives based on Students? Learning Styles and Cognitive Styles. *Educational Technology & Society*, 16(4):185–200.

Zakaria, M. R., Moore, A., Moore, A., Ashman, H., Ashman, H. L., Stewart, C., Stewart, C. D., Brailsford, T., and Brailsford, T. J. (2002). The hybrid model for adaptive educational hypermedia. *LECTURE NOTES IN COMPUTER SCIENCE*, 2347/2006,:580–585.

Zhang, C., Zhao, R., and Zhou, Z. (2010). An Agent-based Architecture for E-Learning System. In *The 2nd International Workshop on Intelligent Systems and Applications (ISA)*, pages 1–4. IEEE.

Zhi-xue, W., Xin, J., Qing-chao, D., Hong-yue, H., and Qing-long, W. (2012). ECA rule modeling language based on UML. In *International Conference on Computer Science and Automation Engineering (CSAE)*, volume 1, pages 623–628. IEEE.

Zimmermann, M. (2011). Adaption of multimedia e-learning services to mobile environments. In *Global Engineering Education Conference (EDUCON)*, pages 671–678. IEEE.

Appendix A

Design Aspects

A.1 The Learning Styles Questionnaire in Moodle

The learning styles questionnaire has been designed and developed in Moodle as a feedback activity. This questionnaire has been adapted from the Index of Learning Styles questionnaire developed by Felder and Soloman. Each learner must complete the questionnaire in order to see the adaptation effect. This questionnaire is offered at course level so that the learner can complete it whenever required. When the learner submits the questionnaire, the learner is directed to the adaptive course page based on the results of his/her learning styles preferences. However, the learner can see the course page without any adaptation effects at any time. The following figures, Figure A.1 and Figure A.2, present the design of the learning styles questionnaire in Moodle.

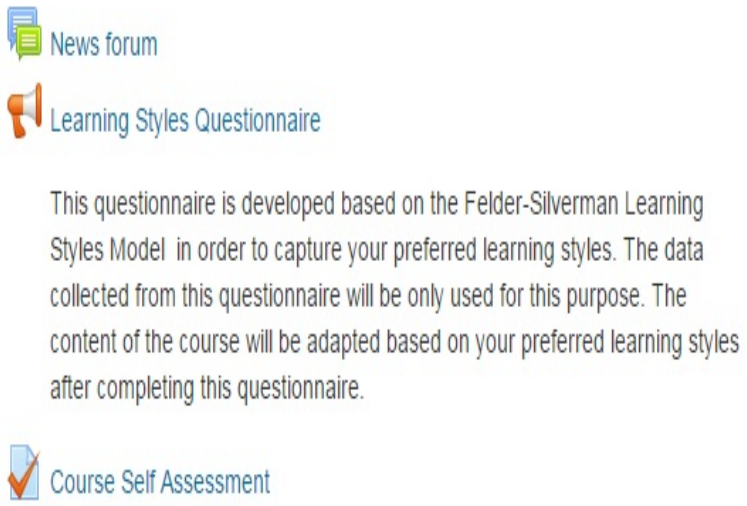


FIGURE A.1: The learning styles questionnaire activity

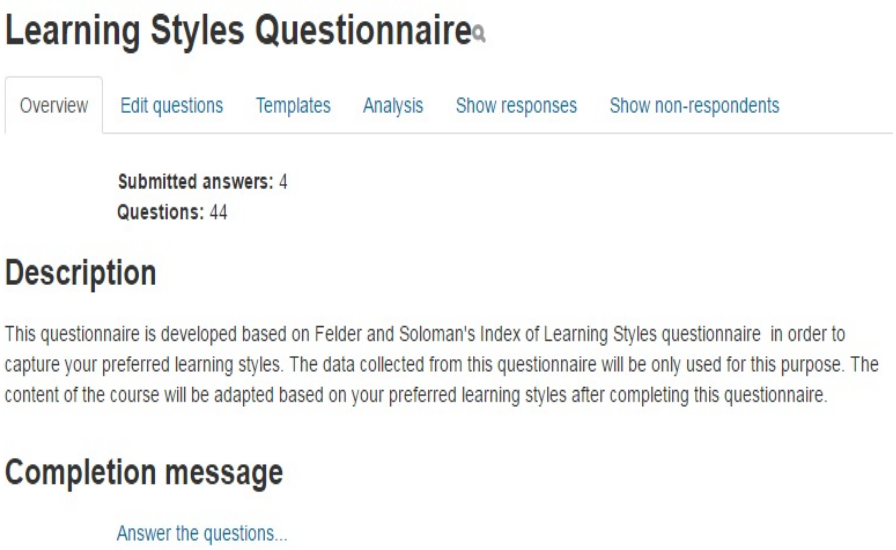
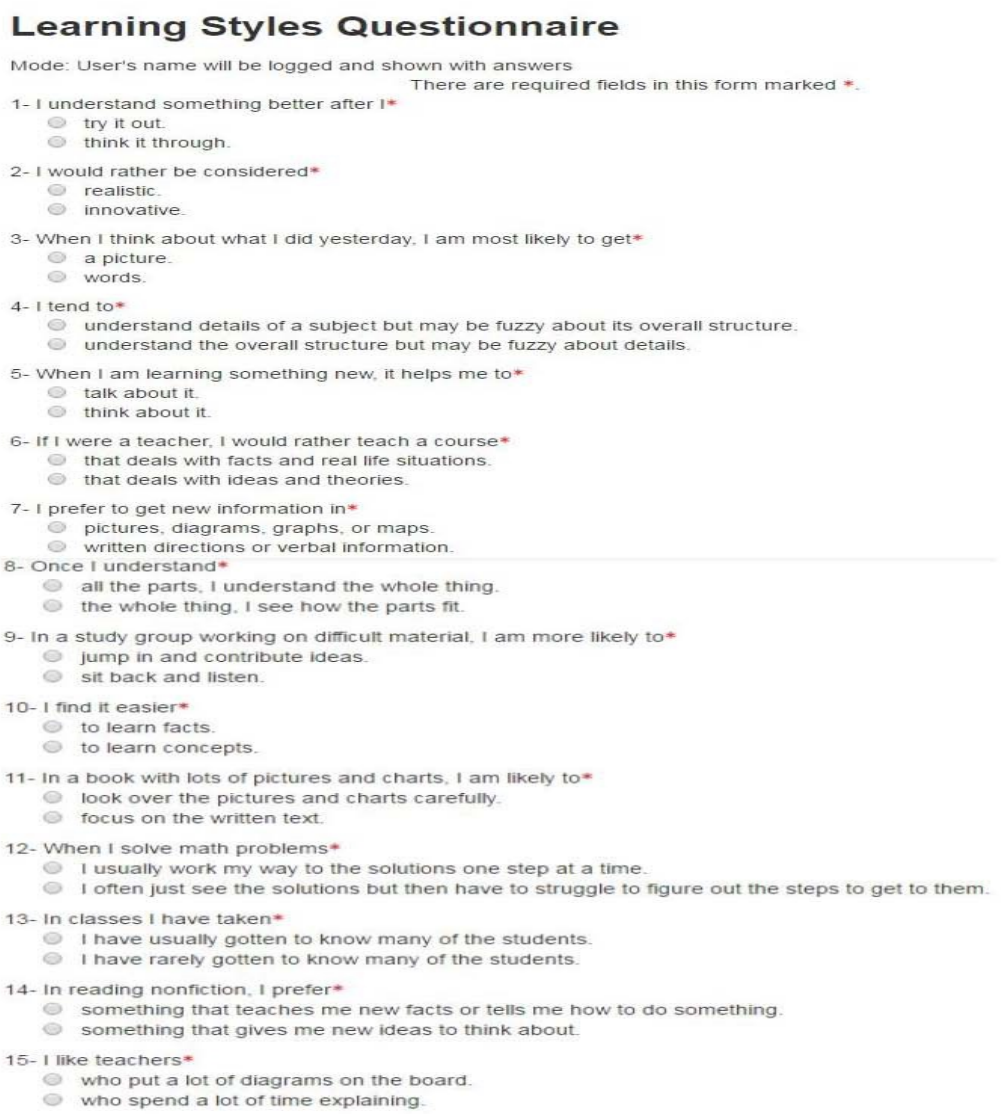


FIGURE A.2: The structure of learning styles questionnaire in Moodle

The learning styles questionnaire has 44 questions. Each question examines a particular learning style preference in the four dimensions of the FSLSM

as discussed in Section 5.2.1. Figures A.3, A.4 and A.5 are snapshots of the learning styles questions as presented to learners in Moodle:



Learning Styles Questionnaire

Mode: User's name will be logged and shown with answers

There are required fields in this form marked *.

1- I understand something better after I*
- try it out.
- think it through.

2- I would rather be considered*
- realistic.
- innovative.

3- When I think about what I did yesterday, I am most likely to get*
- a picture.
- words.

4- I tend to*
- understand details of a subject but may be fuzzy about its overall structure.
- understand the overall structure but may be fuzzy about details.

5- When I am learning something new, it helps me to*
- talk about it.
- think about it.

6- If I were a teacher, I would rather teach a course*
- that deals with facts and real life situations.
- that deals with ideas and theories.

7- I prefer to get new information in*
- pictures, diagrams, graphs, or maps.
- written directions or verbal information.

8- Once I understand*
- all the parts, I understand the whole thing.
- the whole thing, I see how the parts fit.

9- In a study group working on difficult material, I am more likely to*
- jump in and contribute ideas.
- sit back and listen.

10- I find it easier*
- to learn facts.
- to learn concepts.

11- In a book with lots of pictures and charts, I am likely to*
- look over the pictures and charts carefully.
- focus on the written text.

12- When I solve math problems*
- I usually work my way to the solutions one step at a time.
- I often just see the solutions but then have to struggle to figure out the steps to get to them.

13- In classes I have taken*
- I have usually gotten to know many of the students.
- I have rarely gotten to know many of the students.

14- In reading nonfiction, I prefer*
- something that teaches me new facts or tells me how to do something.
- something that gives me new ideas to think about.

15- I like teachers*
- who put a lot of diagrams on the board.
- who spend a lot of time explaining.

FIGURE A.3: The learning styles questions: snapshot A

16- When I'm analyzing a story or a novel*
- I think of the incidents and try to put them together to figure out the themes.
- I just know what the themes are when I finish reading and then I have to go back and find the incidents that demonstrate them.

17- When I start a homework problem, I am more likely to*
- start working on the solution immediately.
- try to fully understand the problem first.

18- I prefer the idea of*
- certainty.
- theory.

19- I remember best*
- what I see.
- what I hear.

20- It is more important to me that an instructor*
- lay out the material in clear sequential steps.
- give me an overall picture and relate the material to other subjects.

21- I prefer to study*
- in a study group.
- alone.

22- I am more likely to be considered*
- careful about the details of my work.
- creative about how to do my work.

23- When I get directions to a new place, I prefer*
- a map.
- written instructions.

24- I learn*
- at a fairly regular pace. If I study hard, I'll "get it."
- in fits and starts. I'll be totally confused and then suddenly it all "clicks."

25- I would rather first*
- try things out.
- think about how I'm going to do it.

26- When I am reading for enjoyment, I like writers to*
- clearly say what they mean.
- say things in creative, interesting ways.

27- When I see a diagram or sketch in class, I am most likely to remember*
- the picture.
- what the instructor said about it.

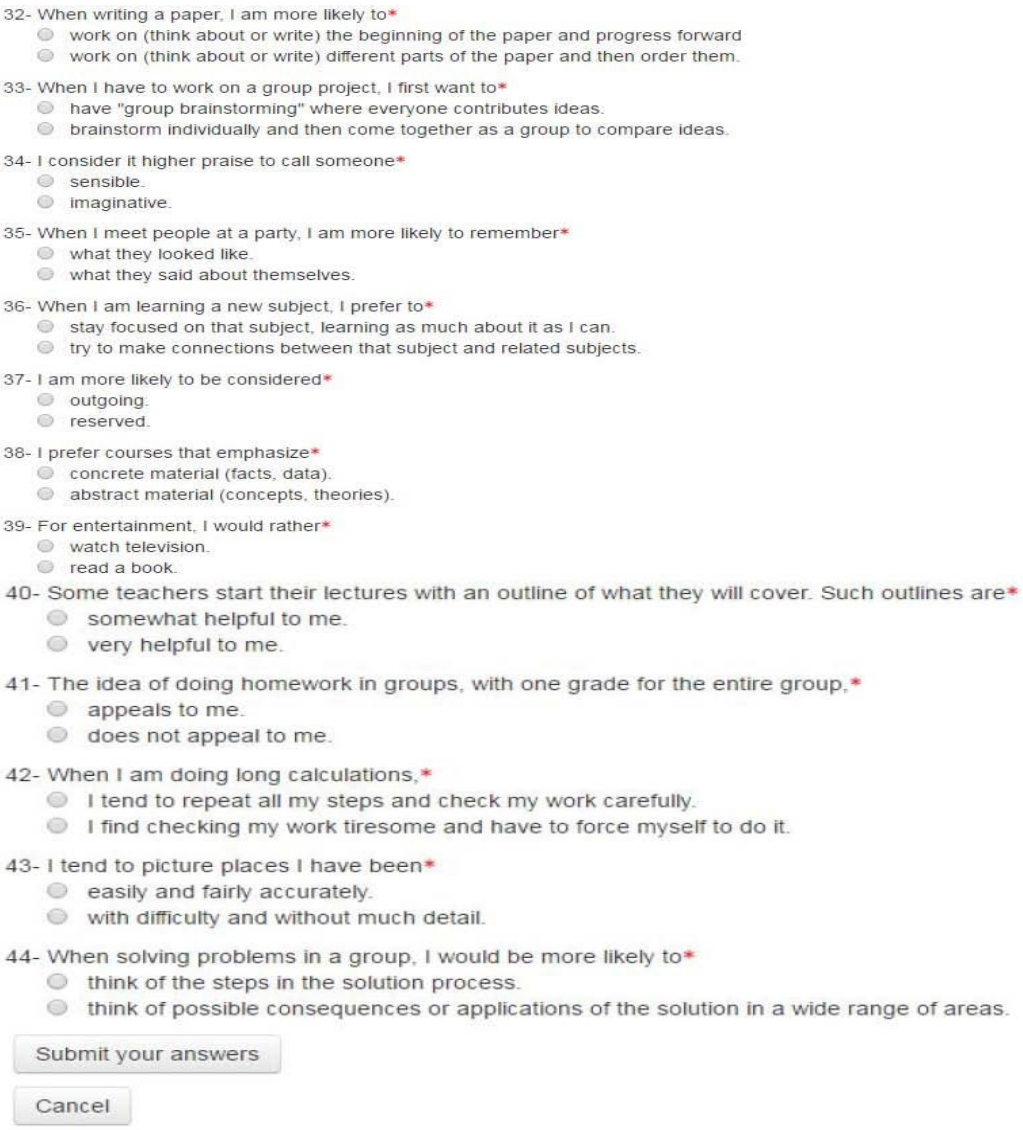
28- When considering a body of information, I am more likely to*
- focus on details and miss the big picture.
- try to understand the big picture before getting into the details.

29- I more easily remember*
- something I have done.
- something I have thought a lot about.

30- When I have to perform a task, I prefer to*
- master one way of doing it.
- come up with new ways of doing it.

31- When someone is showing me data, I prefer*
- charts or graphs.
- text summarizing the results.

FIGURE A.4: The learning styles questions: snapshot B


32- When writing a paper, I am more likely to*
- work on (think about or write) the beginning of the paper and progress forward
- work on (think about or write) different parts of the paper and then order them.

33- When I have to work on a group project, I first want to*
- have "group brainstorming" where everyone contributes ideas.
- brainstorm individually and then come together as a group to compare ideas.

34- I consider it higher praise to call someone*
- sensible.
- imaginative.

35- When I meet people at a party, I am more likely to remember*
- what they looked like.
- what they said about themselves.

36- When I am learning a new subject, I prefer to*
- stay focused on that subject, learning as much about it as I can.
- try to make connections between that subject and related subjects.

37- I am more likely to be considered*
- outgoing.
- reserved.

38- I prefer courses that emphasize*
- concrete material (facts, data).
- abstract material (concepts, theories).

39- For entertainment, I would rather*
- watch television.
- read a book.

40- Some teachers start their lectures with an outline of what they will cover. Such outlines are*
- somewhat helpful to me.
- very helpful to me.

41- The idea of doing homework in groups, with one grade for the entire group,*
- appeals to me.
- does not appeal to me.

42- When I am doing long calculations,*
- I tend to repeat all my steps and check my work carefully.
- I find checking my work tiresome and have to force myself to do it.

43- I tend to picture places I have been*
- easily and fairly accurately.
- with difficulty and without much detail.

44- When solving problems in a group, I would be more likely to*
- think of the steps in the solution process.
- think of possible consequences or applications of the solution in a wide range of areas.

Submit your answers

Cancel

FIGURE A.5: The learning styles questions: snapshot C

A.2 The Design of the Multi-agent Module

The multi-agent module has been designed and developed in JADE. The module has four agents namely, Control Agent, Learner Agent, Adaptive Agent and Course Structure Agent. Each agent has a specific role in the multi-agent structure as discussed in Section 4.3.4.2. This role is associated with the agent's behaviour. The agent's behaviour includes all tasks that must be accomplished with the cooperation of other agents. The class diagram of the multi-agent module is depicted in Figure A.6.

A.3 The Design of the "Shared" Database

The "Shared" database is the main repository of the multi-agent system. It logs all of the required events from the LMS in a timely manner using database triggers. Also, it includes the requirements of the adaptation process, which can be dynamically updated by the course designers and instructors. This database is one of the core components of the proposed architecture discussed in Chapter 4. The schema design of the "Shared" database consists of several database tables, as previously illustrated in Figure 4.7 in Chapter 4. Table A.1 presents a description of each table in the "Shared" database.

A.4 The Implementation of Database Triggers

The technique of database triggers is used in order to simulate the concept of the ECA model. As stated earlier in Chapter 5, the required database triggers should be integrated in the database of the LMS (i.e. Moodle). A snapshot of the database tables of Moodle is depicted in Figure A.7. The implementation

of the database triggers used in order to track the required events in the LMS and to communicate with the multi-agent system is presented in the following listings.

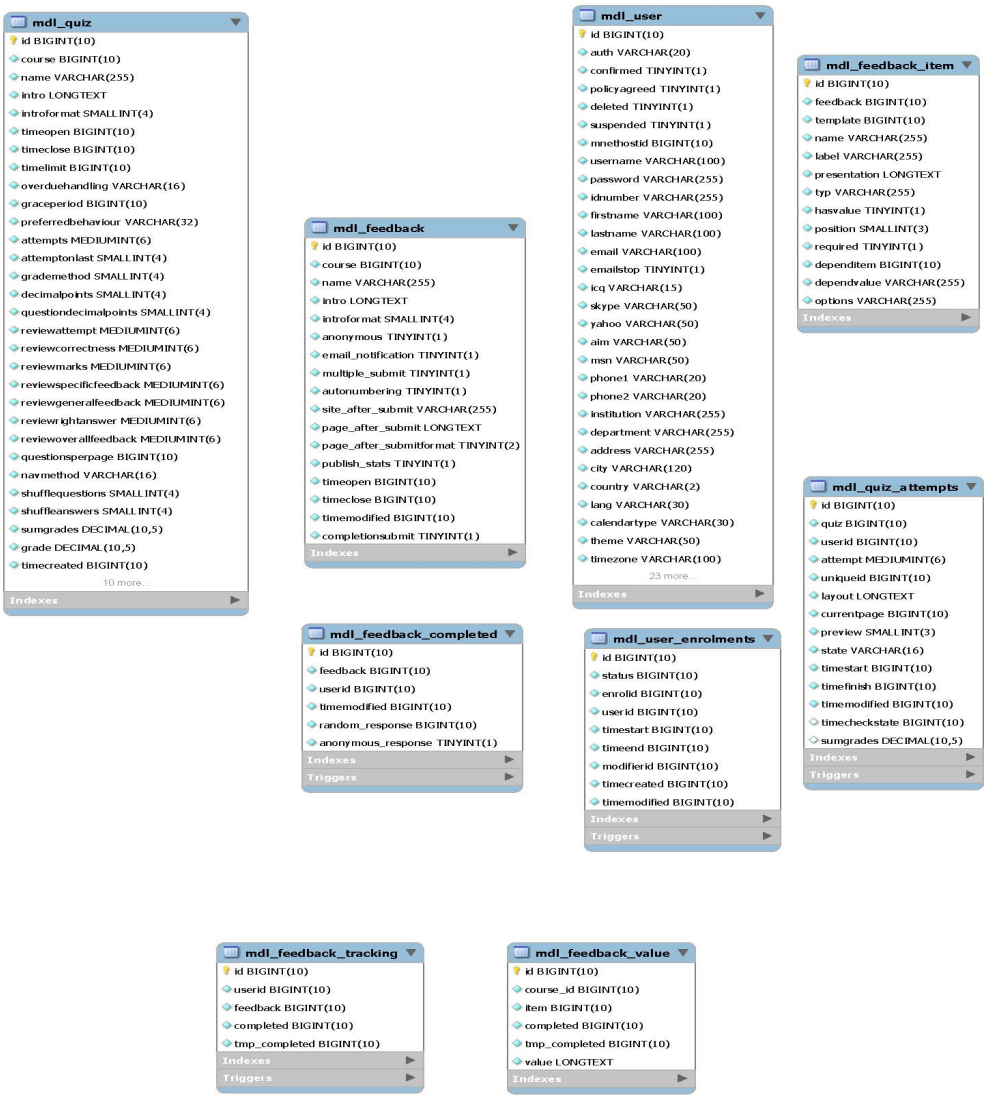


FIGURE A.7: A snapshot of schema design in Moodle

LISTING A.1: The structure of the NewEnrolmentTrigger.sql

```
1 CREATE DEFINER=`root`@`localhost` TRIGGER `moodle`.`NEW_ENROLMENTS_TRIGGER` AFTER
      INSERT ON `mdl_user_enrolments` FOR EACH ROW
```

```
BEGIN
insert into `shareddb`.`std_personal_info`(`std_id`,`course_id`,`enroll_date`) values
    (new.userid,
(select courseid from mdl_enrol where id=new.enrolid),sysdate());

END
```

LISTING A.2: The structure of the QuestionnairesResponsesTrigger.sql

```
CREATE DEFINER=`root`@`localhost` TRIGGER `moodle`.`QUESTIONNAIRES_RESPONSES_TRIGGER`
    AFTER INSERT ON `mdl_feedback_tracking` FOR EACH ROW
BEGIN
Declare feedback_name varchar(255);
Declare quest_rule varchar(255);
if (new.completed>0)
then
select mdl_feedback.name
into feedback_name
from moodle.mdl_feedback
where id=new.feedback;
SELECT
    rule_value
INTO quest_rule FROM
    shareddb.predefined_pedagogical_rules
WHERE
    rule_id = 1;
if(feedback_name = quest_rule) then
INSERT INTO shareddb.std_questionnaire_responses
select new.completed, a.userid, b.course, d.label, replace(c.value,2,-1), sysdate()
from `moodle`.`mdl_feedback_completed` a, `moodle`.`mdl_feedback` b, `moodle`.`
    mdl_feedback_value` c,
`moodle`.`mdl_feedback_item` d
where b.id = a.feedback and c.item=d.id and d.feedback= a.feedback and
 a.id = c.completed and a.userid= new.userid and c.completed= new.completed;
 insert into shareddb.std_completed_questionnaires(id,std_id,course_id,completed_date)
values(new.completed,new.userid, (SELECT course
 FROM moodle.mdl_feedback
 where id=new.feedback), sysdate());
 end if;
```

```
 end if;
END
```

LISTING A.3: The structure of the QuizAttemptsTrigger.sql

```
CREATE DEFINER=`root`@`localhost` TRIGGER `moodle`.`QUIZ_ATTEMPTS_TRIGGER` AFTER
    UPDATE ON `mdl_quiz_attempts` FOR EACH ROW
BEGIN
DECLARE ID BIGINT(10);
DECLARE Quiz_Min VARCHAR(20);
DECLARE Quiz_name VARCHAR(255);
DECLARE QUIZ_ORIG_NAME VARCHAR(255);
SELECT
    MDL_QUIZ.NAME
INTO QUIZ_ORIG_NAME FROM
    MOODLE.MDL_QUIZ
WHERE
    MDL_QUIZ.ID = NEW.QUIZ;
IF(NEW.STATE='finished') THEN
SELECT RULE_VALUE
INTO QUIZ_NAME
FROM SHAREDDB.PREDEFINED_PEDAGOGICAL_RULES
WHERE RULE_ID= 3;
IF(QUIZ_ORIG_NAME = QUIZ_NAME) THEN
SELECT RULE_VALUE
INTO QUIZ_MIN
FROM SHAREDDB.PREDEFINED_PEDAGOGICAL_RULES
WHERE RULE_ID= 2;
IF ( NEW.SUMGRADES <= QUIZ_MIN) THEN
INSERT INTO shareddb.std_quiz_responses
VALUES(ID, NEW.USERID, NEW.QUIZ,(SELECT course FROM moodle.mdl_quiz WHERE mdl_quiz.id=
    new.quiz), NEW.SUMGRADES);
END IF;
END IF;
END IF;
END
```

LISTING A.4: The structure of the CompletedQuestionnaireTrigger.sql

```
CREATE DEFINER=`root`@`localhost` TRIGGER `shareddb`.`COMPLETED_QUESTIONNAIRES_TRIGGER
    ` AFTER INSERT ON `std_completed_questionnaires` FOR EACH ROW
BEGIN
 DECLARE cmd TEXT;
 DECLARE result int(10);
SET cmd=CONCAT('c:/test.bat ',concat(new.id,new.std_id),concat(' ',new.std_id),concat(
    ' ',new.course_id),concat(' ',new.completed_date),concat(' ',new.id));
SET result = sys_exec(cmd);
END
```

LISTING A.5: The structure of the CompletedQuizTrigger.sql

```
CREATE DEFINER=`root`@`localhost` TRIGGER `shareddb`.`completed_quiz_trigger` AFTER
    INSERT ON `std_quiz_responses` FOR EACH ROW
BEGIN
SELECT concat("To: ",get_std_email(new.std_id)),
                "From: ALMS@aou.edu.jo",
                concat("Subject: ",get_std_id(new.std_id)," : Your current progress"),
                "",
                concat("Dear ",get_std_id(new.std_id)," :\n","Kindly be informed that
                    your score in "
                ,get_course_name(new.course_id), " is lower than the required score.
                    If you are facing difficulties doing this course, it is highly
                    recommended that you contact your instructor for further guidance.
                    "
                )
INTO OUTFILE 'c:/maildirectory/mail.eml'
FIELDS TERMINATED BY '\r\n';
END
```

FIGURE A.6: The class diagram of the multi-agent system

TABLE A.1: The description of tables in the "Shared" database

Table Name	Table Description
std_personal_info	This table stores learners' course enrolments
learning_styles	This tables defines the learning styles of the FSLSM
learning_styles_categories	This table identifies the four dimensions of the FSLSM
content_categories	The course content types are defined in this table
content_ls_matching	This table stores the supportive content type for each LS
content_sorting_exceptions	The exceptions of sorting contents are identified in this table
course_content	This table includes the course content specified in the LMS
predefined_pedagogical_rules	In this table, course designers can specify rules of the adaptation process
std_completed_questionnaires	This table logs learners' completed questionnaires
std_content_matching	This table stores the adaptive content that is recommended for each learner
std_ls_results	In this table, the learning styles of each learner are stored
std_questionnaire_responses	This table archives learners' responses for each question in the questionnaire
std_quiz_responses	This table stores the results of course self assessments to provide feedback to learners

Appendix B

Prototype Source Code

This appendix presents the source code of the proposed system prototype written in Java. This code can act as a reference for other researchers who might be interested in developing multi-agent systems in the context of e-learning. The system prototype has been developed in order to validate our hybrid approach. However, the development of the prototype has not reached a level whereby it can be commercially utilised. The source code is illustrated in the following listings.

LISTING B.1: The structure of ControlAgent.java

```
1  package com.omari.agents;
2  import com.omari.behaviors.InformBehavior;
3  import jade.content.lang.sl.SLCodec;
4  import jade.core.Agent;
5  /**
6   * @author Mohammad K. Al-Omari <mohammed.al-omari@dmu.ac.uk>
7   */
8  public class ControlAgent extends Agent {
9      private static final long serialVersionUID = 5149520593242087670L;
10     private InformBehavior informBehavior;
11     public void setup() {
12         getContentManager().registerLanguage(new SLCodec());
13         addBehaviour(informBehavior);
14     }
```

```
15      public void setInformBehavior(InformBehavior informBehavior) {
16          this.informBehavior = informBehavior;
17      }
18  }
```

LISTING B.2: The structure of InformBehavior.java

```
1   package com.omari.behaviors;
2   import com.omari.constants.Constants;
3   import jade.content.lang.sl.SLCodec;
4   import jade.core.AID;
5   import jade.core.Agent;
6   import jade.core.behaviours.CyclicBehaviour;
7   import jade.lang.acl.ACLMessage;
8   import java.util.UUID;
9   /**
10   * Cyclic behaviour class to let the queue know that we received a message
11   */
12  public class InformBehavior extends CyclicBehaviour {
13      private static final long serialVersionUID = -357609527755061303 1L;
14      private final Agent agent;
15      public InformBehavior(Agent a) {
16          super(a);
17          this.agent = a;
18      }
19      public void action() {
20          try {
21              ACLMessage msg = agent.blockingReceive();
22              String from = msg.getSender().getLocalName();
23              switch (msg.getPerformative()) {
24                  case Constants.MISSION_DONE:
25                      System.out.println("Agent [" + this.agent.getLocalName() + "]
                            received [MISSION_DONE] from [" + from + "] with conversation
                            Id: " + msg.getConversationId());
26                      sendConfirmMessageToAgent(from);
27                      break;
28                  case ACLMessage.CONFIRM:
29                      System.out.println("Agent [" + this.agent.getLocalName() + "]
                            received [CONFIRM] from [" + from + "] with conversation Id: "
                             + msg.getConversationId());
```

```
                    break;
                default:
                    throw new RuntimeException("Unknown Message: " + ACLMessage.
                        getPerformative(msg.getPerformative()));
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
    private void sendConfirmMessageToAgent(String receiverAgentName) throws Exception
        {
        ACLMessage msg = new ACLMessage(ACLMessage.CONFIRM);
        AID receiver = new AID(receiverAgentName, false);
        msg.setSender(this.agent.getAID());
        msg.addReceiver(receiver);
        msg.setConversationId(UUID.randomUUID().toString());
        msg.setEncoding("UTF-8");
        msg.setPostTimeStamp();
        msg.setLanguage(new SLCodec().getName());
        System.out.println("Agent [" + this.agent.getLocalName() + "] sending [
            MISSION_DONE_CONFIRM] to [" + receiverAgentName + "] with conversation Id:
             " + msg.getConversationId());
        this.agent.send(msg);
    }
}
```

LISTING B.3: The structure of LearnerAgent.java

```
package com.omari.agents;
import com.omari.behaviors.LearnerBehavior;
import jade.content.lang.sl.SLCodec;
import jade.core.Agent;
/**
 * @author Mohammad K. Al-Omari <mohammed.al-omari@dmu.ac.uk>
 */
public class LearnerAgent extends Agent {
    private static final long serialVersionUID = -5362894968726288045L;
    private LearnerBehavior learnerBehavior;
    public void setup() {
        getContentManager().registerLanguage(new SLCodec());
```

```
13          addBehaviour(learnerBehavior);
14      }
15      public void setLearnerBehavior(LearnerBehavior learnerBehavior) {
16          this.learnerBehavior = learnerBehavior;
17      }
18  }
```

LISTING B.4: The structure of LearnerBehavior.java

```
1  package com.omari.behaviors;
2  import com.fasterxml.jackson.annotation.JsonInclude;
3  import com.fasterxml.jackson.databind.ObjectMapper;
4  import com.omari.database.DataAccessObject;
5  import com.omari.model.CompletedQuestionnaire;
6  import com.omari.model.LearningStyleCategory;
7  import com.omari.model.LearningStyleResult;
8  import com.omari.model.StudentResponse;
9  import jade.content.lang.sl.SLCodec;
10 import jade.content.onto.BasicOntology;
11 import jade.core.AID;
12 import jade.core.Agent;
13 import jade.core.behaviours.CyclicBehaviour;
14 import jade.lang.acl.ACLMessage;
15 import java.util.*;
16 /**
17  * Cyclic behavior to watch for messages from the Inform Behavior
18  */
19 public class LearnerBehavior extends CyclicBehaviour {
20     private static final long serialVersionUID = 8030725090627048175L;
21     private final Agent agent;
22     private DataAccessObject dataAccessObject;
23     private static final ObjectMapper objectMapper = new ObjectMapper();
24     private static EnumMap<LearningStyleCategory, List<Integer>>
           learningStyleCategoryListEnumMap = new EnumMap<LearningStyleCategory, List<
           Integer>>(LearningStyleCategory.class);
25     static {
26         learningStyleCategoryListEnumMap.put(LearningStyleCategory.ACT_REF, new
               ArrayList<Integer>(Arrays.asList(1, 5, 9, 13, 17, 21, 25, 29, 33, 37, 41))
               );
```

```
27          learningStyleCategoryListEnumMap.put(LearningStyleCategory.SEN_INT, new
                ArrayList<Integer>(Arrays.asList(2, 6, 10, 14, 18, 22, 26, 30, 34, 38, 42)
                ));
28          learningStyleCategoryListEnumMap.put(LearningStyleCategory.VIS_VRB, new
                ArrayList<Integer>(Arrays.asList(3, 7, 11, 15, 19, 23, 27, 31, 35, 39, 43)
                ));
29          learningStyleCategoryListEnumMap.put(LearningStyleCategory.SEQ_GLO, new
                ArrayList<Integer>(Arrays.asList(4, 8, 12, 16, 20, 24, 28, 32, 36, 40, 44)
                ));
30
31          objectMapper.setSerializationInclusion(JsonInclude.Include.NON_NULL);
32      }
33      public LearnerBehavior(Agent a) {
34          super(a);
35          this.agent = a;
36      }
37      public void action() {
38          try {
39              ACLMessage msg = agent.blockingReceive();
40              String from = msg.getSender().getLocalName();
41              switch (msg.getPerformative()) {
42                  case ACLMessage.INFORM:
43                      System.out.println("Agent [" + this.agent.getLocalName() + "]
                            received [INFORM] from [" + from + "] with conversation Id: "
                            + msg.getConversationId());
44                      // save content
45                      CompletedQuestionnaire completedQuestionnaire = objectMapper.
                            readValue((String) msg.getContentObject(),
                            CompletedQuestionnaire.class);
46                      Integer receivedAttemptId = completedQuestionnaire.getAttemptId();
47                      // get all responses from database
48                      List<StudentResponse> studentResponses = dataAccessObject.
                            getStudentResponses(receivedAttemptId);
49                      System.out.println("Agent [" + this.agent.getLocalName() + "]
                            retrieved the following data from database:");
50                      for (StudentResponse studentResponse : studentResponses) {
51                          System.out.println(studentResponse.getAttemptId() + ", " +
                                studentResponse.getStudentNumber() + ", " +
52                                  studentResponse.getCourseNumber() + ", " +
                                        studentResponse.getResponseId() + ", "
53                                  + studentResponse.getResponseValue() + ", " +
                                        studentResponse.getResponseDate());
```

```
54                  }
55                  // send CONFIRM back to sender
56                  sendConfirmMessageToAgent(from);
57                  List<LearningStyleResult> styleResults = calculateLearningStyles(
                        studentResponses);
58                  msg = new ACLMessage(ACLMessage.INFORM);
59                  AID receiver = new AID("AdaptiveAgent", false);
60                  msg.setSender(this.agent.getAID());
61                  msg.addReceiver(receiver);
62                  msg.setConversationId(UUID.randomUUID().toString());
63                  msg.setEncoding("UTF-8");
64                  msg.setPostTimeStamp();
65                  msg.setOntology(BasicOntology.STRING);
66                  msg.setLanguage(new SLCodec().getName());
67                  msg.setContentObject(objectMapper.writeValueAsString(styleResults)
                        );
68                  System.out.println("Agent [" + this.agent.getLocalName() + "]
                        sending styleResults to [" + ((AID) msg.getAllReceiver().next
                        ()).getLocalName() + "] with conversation Id: " + msg.
                        getConversationId());
69                  this.agent.send(msg);
70                  break;
71              case ACLMessage.CONFIRM:
72                  System.out.println("Agent [" + this.agent.getLocalName() + "]
                        received [CONFIRM] from [" + from + "] with conversation Id: "
                         + msg.getConversationId());
73                  break;
74              default:
75                  throw new RuntimeException("Unknown Message: " + ACLMessage.
                        getPerformative(msg.getPerformative()));
76          }
77      } catch (Exception e) {
78          e.printStackTrace();
79      }
80  }
81  private List<LearningStyleResult> calculateLearningStyles(List<StudentResponse>
        studentResponses) {
82      List<LearningStyleResult> styleResults = new ArrayList<>();
83      for (Map.Entry<LearningStyleCategory, List<Integer>> entry :
            learningStyleCategoryListEnumMap.entrySet()) {
84          LearningStyleResult learningStyleResult = new LearningStyleResult();
85          Integer score = 0;
```

```
86              for (StudentResponse studentResponse : studentResponses) {
87                  if (entry.getValue().contains(studentResponse.getResponseId())) {
88                      score += studentResponse.getResponseValue();
89                  }
90              }
91              learningStyleResult.setCategoryId(entry.getKey().
                    getLearningStyleCategoryId());
92              learningStyleResult.setCourseNumber(studentResponses.get(0).
                    getCourseNumber());
93              learningStyleResult.setStudentNumber(studentResponses.get(0).
                    getStudentNumber());
94              learningStyleResult.setStyleId(score >= -3 ? entry.getKey().getLeft().
                    getLearningStyleId() : entry.getKey().getRight().getLearningStyleId())
                    ;
95              learningStyleResult.setStrength(getStrength(score));
96              styleResults.add(learningStyleResult);
97          }
98          return styleResults;
99      }
100     private String getStrength(Integer score) {
101         if (Math.abs(score) <= 3) {
102             return "Balanced";
103         } else if (Math.abs(score) <= 7) {
104             return "Moderate";
105         } else {
106             return "Strong";
107         }
108     }
109     private void sendConfirmMessageToAgent(String receiverAgentName) throws Exception
            {
110         // send a message of type CONFIRM
111         ACLMessage msg = new ACLMessage(ACLMessage.CONFIRM);
112         AID receiver = new AID(receiverAgentName, false);
113         msg.setSender(this.agent.getAID());
114         msg.addReceiver(receiver);
115         msg.setConversationId(UUID.randomUUID().toString());
116         msg.setEncoding("UTF-8");
117         msg.setPostTimeStamp();
118         msg.setLanguage(new SLCodec().getName());
119         System.out.println("Agent [" + this.agent.getLocalName() + "] sending [CONFIRM
                ] to [" + receiverAgentName + "] with conversation Id: " + msg.
                getConversationId());
```

```
120         this.agent.send(msg);
121     }
122     public void setDataAccessObject(DataAccessObject dataAccessObject) {
123         this.dataAccessObject = dataAccessObject;
124     }
125 }
```

LISTING B.5: The structure of AdaptiveAgent.java

```
1  /*
2   * To change this license header, choose License Headers in Project Properties.
3   * To change this template file, choose Tools | Templates
4   * and open the template in the editor.
5   */
6  package com.omari.agents;
7  import com.omari.behaviors.AdaptiveBehavior;
8  import jade.content.lang.sl.SLCodec;
9  import jade.core.Agent;
10 /**
11  * @author Mohammad K. Al-Omari <mohammed.al-omari@dmu.ac.uk>
12  */
13 public class AdaptiveAgent extends Agent {
14     private static final long serialVersionUID = 9065403504732384245L;
15     private AdaptiveBehavior adaptiveBehavior;
16     public void setup() {
17         getContentManager().registerLanguage(new SLCodec());
18         addBehaviour(adaptiveBehavior);
19     }
20     public void setAdaptiveBehavior(AdaptiveBehavior adaptiveBehavior) {
21         this.adaptiveBehavior = adaptiveBehavior;
22     }
23 }
```

LISTING B.6: The structure of AdaptiveBehavior.java

```
1 package com.omari.behaviors;
2 import com.fasterxml.jackson.annotation.JsonInclude;
3 import com.fasterxml.jackson.core.type.TypeReference;
4 import com.fasterxml.jackson.databind.ObjectMapper;
```

```
5  import com.omari.constants.Constants;
6  import com.omari.database.DataAccessObject;
7  import com.omari.model.*;
8  import jade.content.lang.sl.SLCodec;
9  import jade.content.onto.BasicOntology;
10 import jade.core.AID;
11 import jade.core.Agent;
12 import jade.core.behaviours.CyclicBehaviour;
13 import jade.lang.acl.ACLMessage;
14 import java.util.*;
15 /**
16  * Cyclic behaviour to watch for messages from the learner Behaviour
17  */
18 public class AdaptiveBehavior extends CyclicBehaviour {
19     private static final long serialVersionUID = 8289747131219495275L;
20     private final Agent agent;
21     protected DataAccessObject dataAccessObject;
22     private static final ObjectMapper objectMapper = new ObjectMapper();
23     static {
24         objectMapper.setSerializationInclusion(JsonInclude.Include.NON_NULL);
25     }
26     public AdaptiveBehavior(Agent a) {
27         super(a);
28         this.agent = a;
29     }
30     public void action() {
31         try {
32             ACLMessage msg = agent.blockingReceive();
33             String from = msg.getSender().getLocalName();
34             switch (msg.getPerformative()) {
35                 case ACLMessage.CONFIRM:
36                     System.out.println("Agent [" + this.agent.getLocalName() + "]
                           received [CONFIRM] from [" + from + "] with conversation Id: "
                            + msg.getConversationId());
37                     break;
38                 case ACLMessage.INFORM:
39                     // read content
40                     List<LearningStyleResult> styleResults = objectMapper.readValue((
                           String) msg.getContentObject(), new TypeReference<List<
                           LearningStyleResult>>() {
41                     });
```

```
42                      System.out.println("Agent [" + this.agent.getLocalName() + "]
                            received [" + styleResults.size() + "] style results. with
                            conversation Id: " + msg.getConversationId());
43                      System.out.println("Agent [" + this.agent.getLocalName() + "]
                            received the following results:");
44                      List<String> deletedResults = new ArrayList<>();
45                      for (LearningStyleResult learningStyleResult : styleResults) {
46                          if (!deletedResults.contains(learningStyleResult.
                                getStudentNumber() + "-" + learningStyleResult.
                                getCourseNumber())) {
47                              // delete all previous learning style results
48                              dataAccessObject.deleteAllStudentLearningStyleResults(
                                    learningStyleResult.getStudentNumber(),
49                                      learningStyleResult.getCourseNumber());
50                              deletedResults.add(learningStyleResult.getStudentNumber()
                                    + "-" + learningStyleResult.getCourseNumber());
51                          }
52                          // insert
53                          dataAccessObject.insertResult(learningStyleResult);
54                          System.out.println("Agent [" + this.agent.getLocalName() + "]
                                inserted: " +
55                                  LearningStyleCategory.from(learningStyleResult.
                                        getCategoryId()) + " - " +
56                                  LearningStyle.from(learningStyleResult.getStyleId()) +
                                         " - " + learningStyleResult.getStrength());
57                      }
58                      // send back confirmation to learner agent
59                      sendConfirmMessageToAgent(from);
60                      // send mission done to course structure
61                      msg = new ACLMessage(ACLMessage.INFORM);
62                      AID receiver = new AID("CourseStructureAgent", false);
63                      msg.setSender(this.agent.getAID());
64                      msg.addReceiver(receiver);
65                      msg.setConversationId(UUID.randomUUID().toString());
66                      msg.setEncoding("UTF-8");
67                      msg.setPostTimeStamp();
68                      msg.setOntology(BasicOntology.STRING);
69                      msg.setLanguage(new SLCodec().getName());
70                      msg.setContentObject(objectMapper.writeValueAsString(styleResults)
                            );
```

```
71                  System.out.println("Agent [" + this.agent.getLocalName() + "]
                        sending styleResults to [" + ((AID) msg.getAllReceiver().next
                        ()).getLocalName() + "] with conversation Id: " + msg.
                        getConversationId());
72                  this.agent.send(msg);
73                  break;
74              case Constants.MISSION_DONE_CONFIRM:
75                  System.out.println("Agent [" + this.agent.getLocalName() + "]
                        received [MISSION_DONE_CONFIRM] from [" + from + "] with
                        conversation Id: " + msg.getConversationId());
76                  break;
77              case ACLMessage.INFORM_IF:
78                   CourseContentStyles courseContentStyles = objectMapper.readValue
                         ((String) msg.getContentObject(), new TypeReference<
                         CourseContentStyles>() {
79                  });
80                  // send back confirmation to course structure agent
81                  sendConfirmMessageToAgent(from);
82                  Integer studentNumber = courseContentStyles.getStudentId();
83                  styleResults = courseContentStyles.getStyleResults();
84                  List<ContentLearningStyleMatch> contentLearningStyleMatches =
                        courseContentStyles.getContentLearningStyleMatches();
85                  List<CourseContent> courseContents = courseContentStyles.
                        getCourseContents();
86                  Map<String, String> courseSectionCategoryContentMap = new HashMap
                        <>();
87                  for (CourseContent courseContent : courseContents) {
88                      courseSectionCategoryContentMap.put(courseContent.getCourseId
                            () + "-" + courseContent.getSectionId() + "-" +
                            courseContent.getCategoryId(), courseContent.getContent())
                            ;
89                  }
90                  // prepare content learning style match
91                  Map<String, Map<Integer, Integer>> courseSectionCategoryMap =
                        prepareContentLearningStyleMatches(styleResults,
                        contentLearningStyleMatches);
92                  // get the order of the categories
93                  List<ContentCategory> contentCategories = dataAccessObject.
                        getContentCategoriesInOrder();
94                  boolean style5Exists = false;
95                  String style5Strength = "Strong";
96                  // do the special case for learning style 5
```

```
97                  for (LearningStyleResult learningStyleResult : styleResults) {
98                      if (learningStyleResult.getStyleId().equals(5)) {
99                          style5Exists = true;
100                         style5Strength = learningStyleResult.getStrength();
101                         break;
102                     }
103                 }
104                 String finalVisibleContent = "";
105                 List<Integer> availableCategories = new ArrayList<>();
106                 // if category doesn't exist in the Map, don't include it
107                 for (Map.Entry<String, Map<Integer, Integer>> a :
                        courseSectionCategoryMap.entrySet()) {
108                     for (Map.Entry<Integer, Integer> b : a.getValue().entrySet())
                            {
109                         availableCategories.add(b.getKey());
110                     }
111                 }
112                 for (ContentCategory contentCategory : contentCategories) {
113                     if (!availableCategories.contains(contentCategory.
                            getCategoryId())) {
114                         continue;
115                     }
116                     if (contentCategory.getCategoryId().equals(8)) {
117                         if (style5Exists && style5Strength.equalsIgnoreCase("
                                balanced")) {
118                             finalVisibleContent += "#CATEGORY" + contentCategory.
                                    getCategoryId() + "#,";
119                         }
120                     } else if (contentCategory.getCategoryId().equals(3)) {
121                         if (style5Exists && !style5Strength.equalsIgnoreCase("
                                balanced")) {
122                             finalVisibleContent += "#CATEGORY" + contentCategory.
                                    getCategoryId() + "#,";
123                         }
124                     } else {
125                         finalVisibleContent += "#CATEGORY" + contentCategory.
                                getCategoryId() + "#,";
126                     }
127                 }
128                 // print for testing
129                 System.out.println("Before exceptions, the order of the categories
                        will be: " + finalVisibleContent);
```

```
130                     // do sorting exceptions
131                     finalVisibleContent = doSortingExceptions(styleResults,
                            finalVisibleContent);
132                     System.out.println("After exceptions, the order of the categories
                            will be: " + finalVisibleContent);
133                     // get the content for each category
134                     for (Map.Entry<String, Map<Integer, Integer>> courseSectionEntry :
                             courseSectionCategoryMap.entrySet()) {
135                         try {
136                             Integer courseId = Integer.parseInt(courseSectionEntry.
                                    getKey().split(",")[0]);
137                             Integer sectionId = Integer.parseInt(courseSectionEntry.
                                    getKey().split(",")[1]);
138                             String visibleContentForSection = finalVisibleContent;
139                             // delete from std_content_matching for std_id and
                                    course_id
140                             dataAccessObject.
                                    deleteAllStudentContentMatchingForStudentAndCourse(
                                    studentNumber, courseId);
141                             for (Map.Entry<Integer, Integer> entry :
                                    courseSectionEntry.getValue().entrySet()) {
142                                 String content = courseSectionCategoryContentMap.get(
                                        courseId + "-" + sectionId + "-" + entry.getKey())
                                        .trim();
143                                 System.out.println("Agent [" + this.agent.getLocalName
                                        () + "] processed SUM as: [" + entry.getValue() +
                                        "] for course,section,category:" + courseId
144                                         + "," + sectionId + "," + entry.getKey());
145                                 if (entry.getValue() < 3 &&
146                                         entry.getKey() != 3 && entry.getKey() != 7 &&
                                                entry.getKey() != 8) {
147                                     // take half of the content
148                                     int contentLength = content.split(",").length;
149                                     int halfLength = 0;
150                                     if (contentLength == 1) {
151                                         content = "";
152                                     } else {
153                                         String tempContent = "";
154                                         halfLength = (contentLength - 1) / 2;
155                                         for (int i = 0; i <= halfLength; i++) {
156                                             if (i == halfLength) {
157                                                 tempContent += content.split(",")[i];
```

```
158                                     } else {
159                                         tempContent += content.split(",")[i] +
                                                ",";
160                                     }
161                                 }
162                                 content = tempContent;
163                             }
164                         }
165                         System.out.println("Agent [" + this.agent.getLocalName
                                () + "] processed content as: [" + content + "]
                                for course,section,category:" + courseId
                                    + "," + sectionId + "," + entry.getKey());
166                         // replace the CATEGORY in the visible area
167                         visibleContentForSection = visibleContentForSection.
                                replace("#CATEGORY" + entry.getKey() + "#",
                                content);
168                     }
169                     // do a final replace of categories not processed
170                     for (ContentCategory contentCategory : contentCategories)
                            {
171                         visibleContentForSection = visibleContentForSection.
                                replace("#CATEGORY" + contentCategory.
                                getCategoryId() + "#,", "");
172                     }
173                     // do clean up
174                     while (visibleContentForSection.contains(",,")) {
175                         visibleContentForSection = visibleContentForSection.
                                replace(",,", ",");
176                     }
177                     if (visibleContentForSection.startsWith(",")) {
178                         visibleContentForSection = visibleContentForSection.
                                substring(1);
179                     }
180                     if (visibleContentForSection.endsWith(",")) {
181                         visibleContentForSection = visibleContentForSection.
                                substring(0, visibleContentForSection.length() -
                                1);
182                     }
183                     StudentContentMatching studentContentMatching = new
                            StudentContentMatching(studentNumber, courseId,
                            sectionId, visibleContentForSection);
184                     // insert in the database
```

```
186                             dataAccessObject.insertNewStudentContentMatching(
                                    studentContentMatching);
187                             // log it
188                             System.out.println("Agent [" + this.agent.getLocalName() +
                                     "] inserted visible content as: [" +
                                    visibleContentForSection + "] for student,course,
                                    section:"
189                                         + studentNumber + "," + courseId
190                                         + "," + sectionId);
191                         } catch (NumberFormatException e) {
192                             e.printStackTrace();
193                         }
194                     }
195                     // send mission complete to the
196                     sendMissionDoneMessageToAgent("ControlAgent");
197               break;
198                 default:
199                     throw new RuntimeException("Unknown Message: " + ACLMessage.
                            getPerformative(msg.getPerformative()));
200             }
201         } catch (Exception e) {
202             e.printStackTrace();
203         }
204     }
205     private String doSortingExceptions(List<LearningStyleResult> styleResults, String
        finalVisibleContent) {
206         // Get sorting exceptions
207         List<ContentSortingException> contentSortingExceptions = dataAccessObject.
            getSortingExceptions();
208         for (ContentSortingException contentSortingException :
            contentSortingExceptions) {
209             // does the learning style exist in style results?
210             for (LearningStyleResult learningStyleResult : styleResults) {
211                 if (learningStyleResult.getStyleId().equals(contentSortingException.
                        getLearningStyleId())) {
212                     String[] categoryIds = contentSortingException.getCategoryIdString
                            ().split(",");
213                     String[] beforeCategoryIds = contentSortingException.
                            getBeforeCategory() == null ? null : contentSortingException.
                            getBeforeCategory().split(",");
```

```
214                     String[] afterCategoryIds = contentSortingException.
                            getAfterCategory() == null ? null : contentSortingException.
                            getAfterCategory().split(",");
215                     for (String categoryId : categoryIds) {
216                         if (beforeCategoryIds != null) {
217                             for (String beforeCategoryId : beforeCategoryIds) {
218                                 if (finalVisibleContent.contains("#CATEGORY" +
                                        beforeCategoryId + "#,") && finalVisibleContent.
                                        contains("#CATEGORY" + categoryId + "#,")) {
219                                     finalVisibleContent = finalVisibleContent.replace(
                                            "#CATEGORY" + categoryId + "#,", "");
220                                     int indexOfBeforeCategory = finalVisibleContent.
                                            indexOf("#CATEGORY" + beforeCategoryId + "#,")
                                            ;
221                                     if (indexOfBeforeCategory == 0) {
222                                         finalVisibleContent = "#CATEGORY" + categoryId
                                                + "#," + finalVisibleContent.substring(
                                                indexOfBeforeCategory);
223                                     } else {
224                                         finalVisibleContent = finalVisibleContent.
                                                substring(0, indexOfBeforeCategory) + "#
                                                CATEGORY" + categoryId + "#," +
                                                finalVisibleContent.substring(
                                                indexOfBeforeCategory);
225                                     }
226                                 }
227                             }
228
229                         }
230                         if (afterCategoryIds != null) {
231                             for (String afterCategoryId : afterCategoryIds) {
232                                 if (finalVisibleContent.contains("#CATEGORY" +
                                        afterCategoryId + "#,") && finalVisibleContent.
                                        contains("#CATEGORY" + categoryId + "#,")) {
233                                     finalVisibleContent = finalVisibleContent.replace(
                                            "#CATEGORY" + categoryId + "#,", "");
234                                     int indexOfAfterCategoryStart =
                                            finalVisibleContent.indexOf("#CATEGORY" +
                                            afterCategoryId + "#,");
235                                     int lengthOfCategory = ("#CATEGORY" +
                                            afterCategoryId + "#,").length();
```

```
236                                 int indexOfAfterCategoryEnd =
                                        indexOfAfterCategoryStart + lengthOfCategory;
237                                 if (indexOfAfterCategoryEnd >= finalVisibleContent
                                        .length()) {
238                                     finalVisibleContent = finalVisibleContent.
                                            substring(0, indexOfAfterCategoryEnd) + "#
                                            CATEGORY" + categoryId + "#,";
239                                 } else {
240                                     finalVisibleContent = finalVisibleContent.
                                            substring(0, indexOfAfterCategoryEnd) + "#
                                            CATEGORY" + categoryId + "#," +
                                            finalVisibleContent.substring(
                                            indexOfAfterCategoryEnd);
241                                 }
242                             }
243                         }
244                     }
245                 }
246             }
247         }
248     }
249     return finalVisibleContent;
250 }
251 /**
252  * the map here is keyed by (courseId,sectionId) --> Map (cat_id, sum)
253  *
254  * @param styleResults
255  * @param contentLearningStyleMatches
256  */
257 private Map<String, Map<Integer, Integer>> prepareContentLearningStyleMatches(List
        <LearningStyleResult> styleResults, List<ContentLearningStyleMatch>
        contentLearningStyleMatches) {
258     Map<String, Map<Integer, Integer>> courseSectionCategoryMap = new HashMap<>();
259
260     for (ContentLearningStyleMatch contentLearningStyleMatch :
            contentLearningStyleMatches) {
261         Map<Integer, Integer> sectionCategoryMap = courseSectionCategoryMap.get(
                contentLearningStyleMatch.getCourseNumber() + "," +
                contentLearningStyleMatch.getSectionNumber());
262         if (sectionCategoryMap == null) {
263             sectionCategoryMap = new HashMap<>();
264         }
```

```
265             Integer categoryResult = sectionCategoryMap.get(contentLearningStyleMatch.
                    getCategoryId());
266             if (categoryResult == null) {
267                 categoryResult = 0;
268             }
269             boolean styleResultExists = false;
270             Integer result = 0;
271             for (LearningStyleResult learningStyleResult : styleResults) {
272                 if (learningStyleResult.getCourseNumber().equals(
                        contentLearningStyleMatch.getCourseNumber())
273                         && learningStyleResult.getStyleId().equals(
                            contentLearningStyleMatch.getStyleId())) {
274                     styleResultExists = true;
275                     result = contentLearningStyleMatch.getNumberOfActivities() *
                            translateWeight(learningStyleResult.getStrength());
276                     break;
277                 }
278             }
279             if (!styleResultExists) {
280                 continue;
281             }
282             categoryResult += result;
283             sectionCategoryMap.put(contentLearningStyleMatch.getCategoryId(),
                    categoryResult);
284             courseSectionCategoryMap.put(contentLearningStyleMatch.getCourseNumber() +
                     "," + contentLearningStyleMatch.getSectionNumber(),
                    sectionCategoryMap);
285         }
286         return courseSectionCategoryMap;
287     }
288     private int translateWeight(String strength) {
289         if ("Strong".equalsIgnoreCase(strength)) {
290             return 3;
291         } else if ("Moderate".equalsIgnoreCase(strength)) {
292             return 2;
293         } else if ("Balanced".equalsIgnoreCase(strength)) {
294             return 1;
295         }
296         throw new RuntimeException("Unknown weight");
297     }
298     private void sendConfirmMessageToAgent(String receiverAgentName) throws Exception
            {
```

```
299         // send a message of type CONFIRM
300         ACLMessage msg = new ACLMessage(ACLMessage.CONFIRM);
301         AID receiver = new AID(receiverAgentName, false);
302         msg.setSender(this.agent.getAID());
303         msg.addReceiver(receiver);
304         msg.setConversationId(UUID.randomUUID().toString());
305         msg.setEncoding("UTF-8");
306         msg.setPostTimeStamp();
307         msg.setLanguage(new SLCodec().getName());
308         System.out.println("Agent [" + this.agent.getLocalName() + "] sending [CONFIRM
                ] to [" + receiverAgentName + "] with conversation Id: " + msg.
                getConversationId());
309         this.agent.send(msg);
310     }
311     private void sendMissionDoneMessageToAgent(String receiverAgentName) throws
            Exception {
312         //Constants.MISSION_DONEsend a message of type MISSION_DONE
313         ACLMessage msg = new ACLMessage(ACLMessage.CONFIRM);
314         AID receiver = new AID(receiverAgentName, false);
315         msg.setSender(this.agent.getAID());
316         msg.addReceiver(receiver);
317         msg.setConversationId(UUID.randomUUID().toString());
318         msg.setEncoding("UTF-8");
319         msg.setPostTimeStamp();
320         msg.setLanguage(new SLCodec().getName());
321         System.out.println("Agent [" + this.agent.getLocalName() + "] sending [
                MISSION_DONE] to [" + receiverAgentName + "] with conversation Id: " + msg
                .getConversationId());
322         this.agent.send(msg);
323     }
324     public void setDataAccessObject(DataAccessObject dataAccessObject) {
325         this.dataAccessObject = dataAccessObject;
326     }
327 }
```

LISTING B.7: The structure of CourseStructureAgent.java

```
1 package com.omari.agents;
2 import com.omari.behaviors.CourseStructureBehavior;
3 import jade.content.lang.sl.SLCodec;
```

```
import jade.core.Agent;
/**
 * @author Mohammad K. Al-Omari <mohammed.al-omari@dmu.ac.uk>
 */
public class CourseStructureAgent extends Agent {
    private static final long serialVersionUID = 9065403507732384245L;
    private CourseStructureBehavior courseStructureBehavior;

    public void setup() {
        getContentManager().registerLanguage(new SLCodec());
        addBehaviour(courseStructureBehavior);
    }
    public void setCourseStructureBehavior(CourseStructureBehavior
        courseStructureBehavior) {
        this.courseStructureBehavior = courseStructureBehavior;
    }
}
```

LISTING B.8: The structure of CourseStructureBehavior.java

```
package com.omari.behaviors;
import com.fasterxml.jackson.annotation.JsonInclude;
import com.fasterxml.jackson.core.type.TypeReference;
import com.fasterxml.jackson.databind.ObjectMapper;
import com.omari.constants.Constants;
import com.omari.database.DataAccessObject;
import com.omari.model.ContentLearningStyleMatch;
import com.omari.model.CourseContent;
import com.omari.model.CourseContentStyles;
import com.omari.model.LearningStyleResult;
import jade.content.lang.sl.SLCodec;
import jade.content.onto.BasicOntology;
import jade.core.AID;
import jade.core.Agent;
import jade.core.behaviours.CyclicBehaviour;
import jade.lang.acl.ACLMessage;
import java.util.ArrayList;
import java.util.List;
import java.util.UUID;
/**
```

```
21  * Cyclic behaviour to watch for messages from the Adaptive Behaviour
22  */
23 public class CourseStructureBehavior extends CyclicBehaviour {
24     private static final long serialVersionUID = 8289747131266495275L;
25     private final Agent agent;
26     protected DataAccessObject dataAccessObject;
27     private static final ObjectMapper objectMapper = new ObjectMapper();
28     static {
29         objectMapper.setSerializationInclusion(JsonInclude.Include.NON_NULL);
30     }
31     public CourseStructureBehavior(Agent a) {
32         super(a);
33         this.agent = a;
34     }
35     public void action() {
36         try {
37             ACLMessage msg = agent.blockingReceive();
38             String from = msg.getSender().getLocalName();
39             switch (msg.getPerformative()) {
40                case ACLMessage.INFORM:
41                     // read content
42                     List<LearningStyleResult> styleResults = objectMapper.readValue((
                           String) msg.getContentObject(), new TypeReference<List<
                           LearningStyleResult>>() {
43                     });
44                     // send CONFIRM back to sender
45                     sendConfirmMessageToAgent(from);
46                     System.out.println("Agent [" + this.agent.getLocalName() + "]
                           received [" + styleResults.size() + "] style results. with
                           conversation Id: " + msg.getConversationId());
47                     List<Integer> learningStyleIds = new ArrayList<>();
48                     Integer studentNumber = null;
49                     Integer courseId = null;
50                     for (LearningStyleResult learningStyleResult : styleResults) {
51                         studentNumber = learningStyleResult.getStudentNumber();
52                         courseId = learningStyleResult.getCourseNumber();
53                         learningStyleIds.add(learningStyleResult.getStyleId());
54                     }
55                     if (studentNumber == null) {
56                         System.out.println("Agent [" + this.agent.getLocalName() + "]
                               found null student number... skipping");
57                         break;
```

```
58                  }
59                  // get the content match
60                  List<ContentLearningStyleMatch> contentLearningStyleMatches =
                        dataAccessObject.
                        getContentLearningStyleMatchByLearningStyleIds(
                        learningStyleIds);
61                  // get the course content
62                  List<CourseContent> courseContents = dataAccessObject.
                        getCourseContents(courseId);
63                  CourseContentStyles courseContentStyles = new CourseContentStyles
                        ();
64                  courseContentStyles.setContentLearningStyleMatches(
                        contentLearningStyleMatches);
65                  courseContentStyles.setCourseContents(courseContents);
66                  courseContentStyles.setStudentId(studentNumber);
67                  courseContentStyles.setStyleResults(styleResults);
68                  msg = new ACLMessage(ACLMessage.INFORM_IF);
69                  AID receiver = new AID("AdaptiveAgent", false);
70                  msg.setSender(this.agent.getAID());
71                  msg.addReceiver(receiver);
72                  msg.setConversationId(UUID.randomUUID().toString());
73                  msg.setEncoding("UTF-8");
74                  msg.setPostTimeStamp();
75                  msg.setOntology(BasicOntology.STRING);
76                  msg.setLanguage(new SLCodec().getName());
77                  msg.setContentObject(objectMapper.writeValueAsString(
                        courseContentStyles));
78                  System.out.println("Agent [" + this.agent.getLocalName() + "]
                        sending courseContentStyles to [" + ((AID) msg.getAllReceiver
                        ().next()).getLocalName() + "] with conversation Id: " + msg.
                        getConversationId());
79                  this.agent.send(msg);
80                  break;
81              case ACLMessage.CONFIRM:
82                  System.out.println("Agent [" + this.agent.getLocalName() + "]
                        received [CONFIRM] from [" + from + "] with conversation Id: "
                         + msg.getConversationId());
83                  break;
84              default:
85                  throw new RuntimeException("Unknown Message: " + ACLMessage.
                        getPerformative(msg.getPerformative()));
86          }
```

```
87          } catch (Exception e) {
88              e.printStackTrace();
89          }
90      }
91      private void sendConfirmMessageToAgent(String receiverAgentName) throws Exception
            {
92          // send a message of type CONFIRM
93          ACLMessage msg = new ACLMessage(ACLMessage.CONFIRM);
94          AID receiver = new AID(receiverAgentName, false);
95          msg.setSender(this.agent.getAID());
96          msg.addReceiver(receiver);
97          msg.setConversationId(UUID.randomUUID().toString());
98          msg.setEncoding("UTF-8");
99          msg.setPostTimeStamp();
100         msg.setLanguage(new SLCodec().getName());
101         System.out.println("Agent [" + this.agent.getLocalName() + "] sending [CONFIRM
                ] to [" + receiverAgentName + "] with conversation Id: " + msg.
                getConversationId());
102         this.agent.send(msg);
103     }
104     public void setDataAccessObject(DataAccessObject dataAccessObject) {
105         this.dataAccessObject = dataAccessObject;
106     }
107 }
```