

Université de Montréal

Cooperative agents for enhancing learner practice

Par

Hongtao Chen

Département d'informatique et recherche opérationnelle

Faculté des arts et des sciences

Mémoire présenté à la faculté des études supérieures

En vue de l'obtention du grade de

Maîtrise ès sciences (M.Sc.)

en informatique

Octobre, 2004

© HONGTAO CHEN, 2004



QA

76

U54

2005

V.011

Direction des bibliothèques

AVIS

L'auteur a autorisé l'Université de Montréal à reproduire et diffuser, en totalité ou en partie, par quelque moyen que ce soit et sur quelque support que ce soit, et exclusivement à des fins non lucratives d'enseignement et de recherche, des copies de ce mémoire ou de cette thèse.

L'auteur et les coauteurs le cas échéant conservent la propriété du droit d'auteur et des droits moraux qui protègent ce document. Ni la thèse ou le mémoire, ni des extraits substantiels de ce document, ne doivent être imprimés ou autrement reproduits sans l'autorisation de l'auteur.

Afin de se conformer à la Loi canadienne sur la protection des renseignements personnels, quelques formulaires secondaires, coordonnées ou signatures intégrées au texte ont pu être enlevés de ce document. Bien que cela ait pu affecter la pagination, il n'y a aucun contenu manquant.

NOTICE

The author of this thesis or dissertation has granted a nonexclusive license allowing Université de Montréal to reproduce and publish the document, in part or in whole, and in any format, solely for noncommercial educational and research purposes.

The author and co-authors if applicable retain copyright ownership and moral rights in this document. Neither the whole thesis or dissertation, nor substantial extracts from it, may be printed or otherwise reproduced without the author's permission.

In compliance with the Canadian Privacy Act some supporting forms, contact information or signatures may have been removed from the document. While this may affect the document page count, it does not represent any loss of content from the document.

Université de Montréal
Faculté des études supérieures

Ce mémoire de maîtrise intitulé

Cooperative agents for enhancing learner practice

Présenté par
Hongtao Chen

A été évalué par un jury composé des personnes suivantes :

Président rapporteur :	Max Mignotte
Directeur de recherche :	Claude Frasson
Membre de jury :	Victor Ostromoukhov

Mémoire accepté le : 2 décembre 2004

Sommaire

En permettant de simuler les interactions d'un système de formation à distance, les agents intelligents peuvent fournir un outil puissant pour améliorer la formation en ligne. Un agent fournit un moyen d'implémenter et de simuler les aspects humains de l'interaction de façon plus efficace que les autres méthodes contrôlées par l'ordinateur. De plus, du point de vue architecture, ceci permet plus de flexibilité dans la conception étant donné que les agents sont des objets indépendants de l'environnement d'apprentissage. Ainsi, des environnements tels que CLE (Cooperative Learning Environment) que nous présentons ici, permettent d'étudier l'effet d'un support personnalisé dans un environnement coopératif; il permet aussi d'adapter le processus d'apprentissage à des apprenants et d'organiser automatiquement des groupes d'apprentissage.

Les résultats préliminaires des recherches sur le CLE montrent que des systèmes basés sur plusieurs agents peuvent être utilisés pour améliorer effectivement la pratique chez l'apprenant. En termes d'impacts la création d'environnements basés sur des agents pour examiner permet d'être à la pointe de la recherche sur les agents intelligents et d'explorer de nouveaux paradigmes sur l'apprentissage et la formation en ligne.

Mot-clé : Agent, Système basés sur des agents, Système de formation à distance, Cooperative Learning Environment, Java FAQ.

Abstract

By using intelligent agents to simulate instruction in an online learning environment, agent-based learning environments can supply as a powerful research tool to study online learning improvement. The agent provides a way to implement and simulate the “human” aspect of instruction in a more ecologically valid way than other controlled computer-based methods. Additionally, from an architectural perspective, since agents are independent objects in the learning environment, it allows for more flexibility in research design. In particular, agent-based learning environments, in systems such as CLE (Cooperative Learning Environment), allow for studying the effect of providing a personalized learning support in a cooperative learning environment, can customize the learning process for individual learners, and organize the learning groups automatically.

Preliminary results from the CLE research indicate that multiple agent-based online learning systems can be used to effectively enhance the learner practice. In terms of overall impact, creating agent-based learning environments to study instructional issues is at the leading edge of research integrating intelligent agent with online education, and in exploring new paradigms for researching online teaching and learning.

Keyword: Agent, Agent-Based System, Online Learning System, Cooperative Learning Environment, Java FAQ.

Acknowledgement

I would like to acknowledge many people for helping me during my thesis work. I would especially like to thank my advisor, Professor Claude Frasson, for his generous time and commitment. Throughout my thesis work, he encouraged me to develop independent thinking and research skills. He continually stimulated my analytical thinking and greatly assisted me with scientific writing. Professor Frasson has been a great source of encouragement and inspiration to me. Without his support this dissertation would not have been written.

I owe my wife Wei Li for her encouragement and support, without her help it would have been a far more difficult task to complete my thesis.

Finally, I thank the Department of Computer Science at the University of Montreal for giving me this opportunity and providing me an environment to do research.

Table of content

Sommaire.....	i
Abstract.....	ii
Acknowledgement.....	iii
Table of content.....	iv
List of figures.....	ix
List of tables.....	xi
List of equations	xii
Chapter 1 Introduction.....	1
1.1 Purpose.....	3
1.2 Cooperative learning environment.....	3
1.3 Prototype application	5
1.4 Overview.....	6
1.5 Contribution	7
Chapter 2 Agent and online learning system	8
2.1 Agent.....	8
2.1.1 Definition	9
2.1.2 Characteristic	13
2.1.3 Classification.....	14
2.1.4 Agent and Object	16
2.2 Online learning system	19

2.2.1 Characteristic	20
2.2.2 Agent-based online learning system	21
2.2.3 Multi-agents system	22
2.3 Online learning system model	24
2.4 Summary	26
Chapter 3 Personalized learning support	27
3.1 Personalized learning support	27
3.2 Comparison	27
3.3 Challenges	29
3.4 The Online Learning Ability	31
3.5 PLS design	32
3.6 Personalization Framework	34
3.7 Summary	36
Chapter 4 Cooperative learning environment	37
4.1 Cooperative learning	38
4.1.1 Benefit	38
4.1.2 Elements of Cooperative Learning	38
4.2 Online cooperative learning	40
4.3 Existing online CLE	41
4.3.1 Simulated student agent	42
4.3.2 Computational model of distance learning	42
4.3.3 Multi-agent architecture for cooperative learning	44
4.4 Agents in cooperative learning	45
4.5 Mechanisms	46
4.6 PLE & CLE	48
4.7 Interaction	50

4.8 Overview	51
Chapter 5 Prototype application	53
5.1 Three-tier application.....	53
5.1.1 Definition	53
5.1.2 Two-tier versus three-tier.....	54
5.1.3 Three-tie architecture	55
5.2 System architecture.....	56
5.2.1 Presentation module.....	58
5.2.2 Presentation modules reference	58
5.2.3 Agents' structure	59
5.2.4 Data access structure.....	60
5.3 XML.....	60
5.4 Data structure.....	62
5.4.1 FAQ item and it's related tables	62
5.4.2 Learner profile and its related tables.....	62
5.4.3 FAQ category.....	63
5.5 Summary.....	63
Chapter 6 Methodologies and algorithms.....	65
6.1 Agent's responsibility	65
6.2 Architecture	66
6.3 Problem situation	68
6.4 Problem detecting routine.....	69
6.4.1 Problem agent action's process.....	69
6.4.2 Affective difficulty reaction roles	70
6.4.3 Interpersonal difficulty reaction roles	72
6.5 Learning from Feedback.....	72

6.6 Vector-based similarity weight measure.....	73
6.7 Summary.....	76
Chapter 7 Implementation.....	77
7.1 Run-time environment and development tools.....	77
7.1.1 Development environment.....	77
7.1.2 Database engine.....	79
7.2 Class diagram.....	80
7.2.1 User class.....	80
7.2.2 Item class.....	81
7.2.3 Database helper class.....	82
7.3 Scenario diagram.....	83
7.3.1 Browsing scenario.....	83
7.3.2 Search scenario.....	84
7.3.3 Reading scenario.....	85
7.3.4 Recommending scenario.....	86
7.4 User interface.....	86
7.4.1 Login & Register.....	86
7.4.2 Select the interesting section.....	87
7.4.3 FAQ list.....	87
7.4.4 Section list.....	88
7.4.5 Question & answer.....	89
7.4.6 Search result.....	90
7.4.7 Recommend a reference.....	91
7.4.8 Customized mouse right click menu.....	92
7.5 Examples & Scenarios.....	93
7.6 Summary.....	99



Chapter 8 Conclusion	100
8.1 Conclusion	100
8.2 Future work	102
Bibliography	105

List of figures

Figure 1-1: Cooperative learning environment.....	4
Figure 2-1: Abstract view of an agent in its environment	13
Figure 2-2: Classification of Agents	15
Figure 2-3: Multi-agent system.....	24
Figure 2-4: A Developing Conceptual Framework for Online Learning	25
Figure 4-1: Social learning model.....	44
Figure 4-2: Communication in CLE learning group.....	48
Figure 5-1: Typical three-tier application architecture	54
Figure 5-2: System architecture	57
Figure 5-3: Presentation modules structure	58
Figure 5-4: Presentation modules reference	59
Figure 5-5: Agents structure	59
Figure 5-6: Virtual database interface.....	60
Figure 5-7: FAQ items and it's related	62
Figure 5-8: User table and it's related	63
Figure 6-1: Logical Architecture	66
Figure 7-1: User class and its related.....	81
Figure 7-2: A FAQ item and its related	82
Figure 7-3: Data access helper class	82
Figure 7-4: Browse scenario	84
Figure 7-5: Search scenario	85
Figure 7-6: Reading scenario.....	85
Figure 7-7: Recommending scenario	86
Figure 7-8: Login	87

Figure 7-9: Select the interesting section.....	87
Figure 7-10: FAQ list.....	88
Figure 7-11: Client side agent pops up to display a message	88
Figure 7-12: Section list.....	89
Figure 7-13: FAQ item - question & answer	90
Figure 7-14: Search result.....	91
Figure 7-15: Asking recommendation	91
Figure 7-16: Recommendation web dialog.....	92
Figure 7-17: Customized mouse right click menu.....	92
Figure 7-18: Find definition.....	93
Figure 7-19: Search in Google.....	93
Figure 7-20: Agent suggests the learner to use the search box.....	94
Figure 7-21: Learner's Start Page.....	94
Figure 7-22: Recommend the learner	95
Figure 7-23: Motivation Tips.....	96
Figure 7-24: Searching without result.....	96
Figure 7-25: Search selected text in the web page.....	97
Figure 7-26: Original List.....	98
Figure 7-27: After learner's recommendation.....	98
Figure 7-28: Comment the topic.....	99

List of tables

Table 2-1: Definitions of agent.....	11
Table 2-2: Evolution of programming approaches	18
Table 5-1: XML example.....	61
Table 5-2: FAQ category	63

List of equations

Equation 6-1: Basic distance.....	74
Equation 6-2: Step distance	74
Equation 6-3: Document distance.....	75
Equation 6-4: Closest distance.....	75
Equation 6-5: Continual length.....	75
Equation 6-6: Document weight for query	75

Chapter 1

Introduction

With the growing interconnectedness such as the internet and wireless tools we want distributed computing and multiple computers to be able to cooperate on difficult tasks efficiently. The web has attracted a great deal of consideration as a medium for delivering the distance education, in a synchronous and asynchronous manner [Buraga 2003].

Online learning using agents is a domain in which much progress has recently been realized. Different techniques such as reinforcement learning [Dahlstrom and Wiewiora 2002] [Paletta and Rome 2000], artificial neural networks [Billard and Hayes 1997] or genetic algorithms have been studied extensively and have produced good results. Recently researchers have focused on how agents can perform cooperative tasks, evaluate the actions and improve the learning process. A number of algorithms have been produced to enhance the behaviours and the learning process of agents [Peeters 2003].

One of the active research aspects is to provide personalized learning support in the online learning system. Personalized learning support system analyses the learning style, the learning process and the learning results of the learner to adjust the curriculum of the learning system according to the learner's knowledge level, adapt the selection of learning material like presentations, examples, illustrations, feedback, tests etc. Because the learner's psychological states are not well considered or difficult to gather in the online learning system, a personalized learning support is still a limitation in most researches [Zhang *et al.* 2003].

For instance, a learner might lack of motivation to complete a learning session when he faces to a problem or a learning situation. The main reason is that learners have varying levels of knowledge, learning styles, and needs. Most of the online learning systems that apply the same approach to all learners cannot deal with them as personalized individuals [Razek *et al.* 2002].

In the online learning environment, the difficulties of a personalized learning support include:

- Learning process personalization

All the online learning systems are pre-programmed. Some systems have the ability to adjust the process to a certain extent for the learner, but they still have some kind of pre-defined procedures, structures or frameworks.

This framework could take effect in most of the situations, but it always has the exception situation in real learning circumstances and the percentage of this kind of exception is higher than the researcher's imagination. The flexibility of the learning process is one of the open research issues even now.

- Learning material personalization

In most of the online learning system, learning materials are limited in the arranged documentations. Especially in the self-enclosed individual learning system, the learner cannot obtain any information outside the system, even from the other learners. The learners, in the traditional cooperative learning system, can exchange their experience each other, but all the materials they faced are still the predetermined which certainly cannot meet all the learners' needs. On the other hand, a great quantity of information resource is available on the Internet which might be convenience for the learners. How to make efficient use of this kind of resources still is a question in the field of online learning.

1.1 Purpose

Since the goal of the research is to build a cooperative learning environment (CLE) based on several agents able to help automatically the learner, according to the questions above, our research is concerned to create a personalized learning support system which can enhance the online learning practice through customized learning process and materials by using cooperative agents.

A collection of agents in the system help the learner to constitute a personal learning environment (PLE), in which the learner stores, organizes and comments information about the learning, and from which new requirement for CLE arises.

1.2 Cooperative learning environment

The CLE is the structuring of virtual learning groups so that learners can work together to maximize their own and each other's learning. Based on the learner's interest, learning style and knowledge level etc, learners compose learning groups automatically by the system. Group members focus on the same learning topic and are able to go deep into the topic by communicating each other.

Since the CLE is generated dynamically during the learning session, learners in the online learning system are capable of having their own learning practices. As a result, online learning system can provide dynamic learning process to support the personalized learning through the CLE.

In a cooperative learning environment (CLE), learning practice information, including the learning experience, reference documentation and valuable commentary, are shared via agents in PLE. In this way, agents change their behaviours not only in response to the

learners' learning progress, state and history, but also to other agents' actions and experience.

Following is a sketch about the CLE:

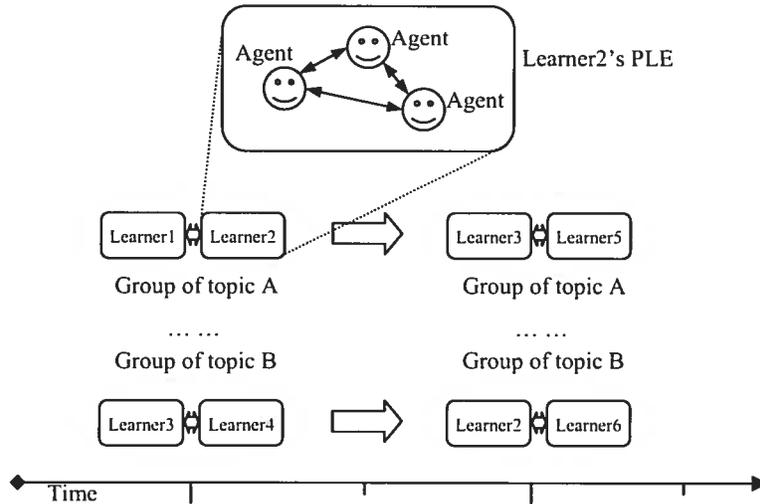


Figure 1-1: Cooperative learning environment

In every learning phase of this environment, learners study in their own PLE which provides learning supports by several learning agents. Agents inside the PLE support the learner in accordance with learner's learning experience; simultaneously they refer to other learner's experience in the learning group to present additional information to the learner.

The learner's learning process is customized by his own knowledge through the agents, but also adapted by the other learner's experience if the agents can detect the significant information from the other learner. For example, if most learners failed on a certain learning question, for learners who can answer it correctly at the first time, they either have enough knowledge background to respond it, or get useful information from the previous learner's commentary in CLE.

1.3 Prototype application

We have developed a prototype application (a Java FAQ helping system) for the use of this framework which we call the cooperative learning environment (CLE). In contrast to the existing models (refer Chapter 4 for detail), it's a combination of the individual learning system and the cooperative learning system. In this application, agents can explicitly communicate each other, either that the action choices of the other agents are directly observable, or that the agents share the learners' status and learning history information. Agents observe all the other agents in the CLE; refine these behaviours with its own experience. They try to detect the information about its potential action capabilities rather than duplicate the behaviours of other agents.

In our CLE, each personal learning environment (PLE) is an independent entity. They are homogeneous and communicate each other in the CLE, and there's no mentor or director in the learning environment. This mechanism makes it possible that the learners in the PLE have their individual learning process based on their knowledge level, personal requirement and learning state. It also resolves the problem in the system which has an expert in the background, and tries to teach the learner by the arranged style and process.

The CLE is composed by the PLEs inside it. Because the agents can provide the extra reference information through the Google web services, the learning materials of the application is not only limited to the prepared documents in the database, but also includes the Google reference which searches documents in more than 1 billions web pages.

Inside the PLE, several agents work together to assist the learner during the learning session. Personal search agent grasps the reference documents from FAQ database and also the Internet by the Google Web Services. Problem agent detects the learner's

weakness and pre-fetches the related information via the search agent. Motivation agent reinforces the learner's motivation by the presentation and dialogue and transfers the result to the problem agent to complete the knowledge.

All the extra reference documents and related information come from the Internet with the Web Services dynamically. Therefore, for different learners with different situation, the reference information could be completely different even with the same question. This mechanism makes the material of the learning system become infinite and possible to personalized material support for the individual learners.

1.4 Overview

In the chapter two, we introduce the basic definitions of the agent and agent-based system. This is needed to have a common view on the subject; we introduce terminology needed for the rest of the thesis and the research situation of agent-based system.

Chapter three discusses online learning, more especially agent-based online learning. Except the benefit of the online learning, we also introduce the agent roles in the learning system and talk about multi-agent learning environments and issues.

Chapter four explains the cooperative agent learning environment, introduces the cooperative learning algorithm. We also present some existing cooperative learning models and discuss the mechanisms and structure of the cooperative learning in that chapter.

Chapter five introduces the prototype application. We focus on the application's architecture, system's structure and data structure which should be used in following chapters for more detail discussion.

Chapter six presents the methodologies and algorithms using our application. We discuss the learning situations analysis and detection and the vector based similarity weight measure which should be used by the agents in our application for the learners' status analysis and the learning process detection.

Chapter seven introduces the implementation of the prototype application. We focus on the class diagrams and the important scenarios of the application. Through the description of the web interfaces, we present some detail technologies used in the application to enhancing the cooperative learning.

In the final chapter we present our conclusions, restate the work done and suggest topics for further research.

1.5 Contribution

In this thesis, we discuss one of the techniques to enhance the online learning process - providing personalized learning support by intelligent agents. We introduce a cooperative learning environment to demonstrate the enhancement of agent-based online learning system. We describe a learner difficulty detecting routine to analyze learners' situation and provide the personalized learning process. We also present a cooperative-based similarity measure algorithm to enhance the search process and result. A prototype application is included in this thesis which implements the learning environment and algorithm.

Agent and online learning system

Our model of cooperative learning environment is based on the framework of multi-agents online learning systems. This chapter briefly reviews some of the most relevant material and concepts within these fields.

2.1 Agent

As we know, in theory, the only thing a computer really can do is binary mathematics. A computer can only work under a series of the orders we designed in advance. If the computer steps into a situation we did not anticipate, the result may become unexpected. For all the stages during the computation, every situation has to be explicitly anticipated and coded by a programmer. This simple fact is at the heart of our relationship with computers.

In the beginning, this behaviour is adequate. What we need the computer to do are just some simple and repeating jobs, such as the mathematic calculation. We assign the expressions and input the arguments, than the computer can always output the expected result. We accept that computers are well-trained, unimaginative labour. However, with the increment of the computation power of the machines, for the appearance of the large number of applications, we require systems that can decide for themselves what they need to do in order to satisfy their design objectives. When we have agents that can operate in rapidly changing, unpredictable, or open environments where there is a significant chance of failure we give them the name of intelligent agents, or also autonomous agents [Peeters 2003].

2.1.1 Definition

Although the term agent is widely used by many people working in closely related areas, there is no single universally accepted definition of an agent. This is because agents can have different degrees of complexity. At one end of the spectrum, agents are extremely simple and have very limited capabilities. Simple agents simulate unintelligent machines such as thermostats, logic gates, and fuses. At the other end, agents have intricate structures and are capable of performing elaborate functions. Complex agents are comparable to biological organisms, complex machines, and people. The lack of a unique definition need not necessarily be a problem; after all, if many people are successfully developing interesting and useful applications, then it hardly matters that they do not agree on terminological details. However, there is also the danger that unless the issue is discussed, “agent” might become a “noise” term, subject to both abuse and misuse, to the potential confusion. Hence, we do not intend to introduce yet another definition for software agent, but our objective is to identify the common characteristics of software agents and explain the concept of agency as related to our framework.

Wooldridge and Jennings [Wooldridge and Jennings 1992] have proposed one of the most comprehensive definitions of agents. They distinguish two general usages of the term “agent”: the first is weak, and relatively un-contentious; the second is stronger, and potentially more contentious.

Weak notion of agency: Perhaps the most general way in which the term agent is used is to denote a hardware or software-based computer system that possesses the following properties:

- **Autonomy:** agents operate without the direct intervention of humans or others, and have some kind of control over their actions and internal state;

- Sociability: agents interact with other agents (and possibly humans) via an interface or possibly some kind of agent-communication language;
- Reactivity: agents perceive their environment, (which may be the physical world, a user via a graphical interface, a collection of other agents, the Internet, or perhaps all of these combined), and respond in a timely fashion to changes that occur in it;
- Pro-activeness: agents do not simply react to their environment; they are able to exhibit goal-directed behaviour by taking the initiative.

In mainstream computer science, the notion of an agent as a self-contained, concurrently executing software process, that encapsulates some states and is able to communicate with other agents via message passing, is seen as a natural development of the object-based concurrent programming paradigm.

The weak notion of agency is also used in agent-based software engineering:

- “[Software agents] communicate with their peers by exchanging messages in an expressive agent communication language. Agents can be as simple as subroutines; but typically they are larger entities with some sort of persistent control.” [Genesereth and Ketchpel 1994]

Stronger notion of agency: For some researchers - particularly those working in Artificial Intelligence (AI) - the term “agent” has a stronger and more specific meaning than that sketched out above. An agent is generally referred to as a computer system that, in addition to having the properties identified above, is either conceptualized or implemented using concepts that are more usually applied to humans. It is quite common in AI to characterize an agent using mentalistic notions, such as knowledge, belief, intention, obligation, or emotion.

Some other typical (and not necessary mutually exclusive) definitions of agents are presented in the table following (Table 2-1) which provide a list of attributes often found in agents: Autonomous, goal-oriented, collaborative, flexible, self-starting, temporal continuity, character, communicative, adaptive, and mobile.

Table 2-1: Definitions of agent

Researchers	Definition
Russel and Norvig	“An agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through effectors”.
Shoham	“An agent is a software entity which functions continuously and autonomously in a particular environment, often inhabited by other agents and processes”.
Kinney, Georgeff, and Rao	“Agents are viewed as having certain mental attitudes, beliefs, desires, and intentions that represent their informational, motivational, and deliberative states, respectively. In BDI (Belief, Desire, and Intention) architecture an agent can be completely specified by the events that it can perceive, the actions it may perform, the beliefs it may hold, the goals it may adopt, and the plans that give rise to its intentions”.
Franklin and Graesser	“An autonomous agent is a system situated within and part of an environment that senses that environment and acts on it, over time, in pursuit of its own agendas and so as to effect what it senses in the future”.
Hayes-Roth	“Intelligent agents continuously perform three functions: perception of dynamic conditions in the environment; action to affect conditions in the environment; and reasoning to intercept perceptions, solve problems, draw inferences, and determine actions”.
Maes	“Autonomous agents are computational systems that inhabit some complex dynamic environment, sense and act autonomously in this environment, and by doing so realize a set of goals or tasks for which they are designed”.
Brustoloni	“Autonomous agents are systems capable of autonomous, purposeful action in the real world. The fact that agents are to perform purposeful action is often interpreted as meaning that they are goal-oriented. A better interpretation is that agents have drives (in a

Researchers	Definition
	somewhat “psychological” sense), and devote their resources to satisfying these drives. In doing so, one can observe purpose in their actions. If the agents are to be autonomous, they have to either know or else be able to find out how to satisfy drives and achieve goals”.
Smith, Cypher, and Spohrer	“An agent is a persistent software entity dedicated to a specific purpose. ‘Persistent’ distinguishes agents from subroutines; agents have their own ideas about how to accomplish tasks, their own agendas. ‘Special purpose’ distinguishes them from multi-function applications; agents are typically much smaller”.
The MuBot Agent	“The term agent is used to represent two orthogonal concepts. The first is the agent’s ability for autonomous execution. The second is the agent's ability to perform domain oriented reasoning”. For example, Microsoft Word's Spelling Assistant is a simple example of an agent. The spelling assistant assists the user of Microsoft Word by autonomously watching over the words that are typed (sensing the environment) and underlines the words that it does not recognize (acting upon the sensed data). The recognition is based on understanding the domain of English words and their spellings.
The IBM Agent	“Intelligent agents are software entities that carry out some set of operations on behalf of a user or another program with some degree of independence or autonomy, and in doing so, employ some knowledge or representation of the user’s goals or desires. Intelligent agents work by allowing people to delegate work that they could have done, to the agent software”
Steiner, Mahling, and Haugeneder	“An agent is the central computational entity, which serves as an explicit model of all entities participating in cooperation. It can be decomposed into the following three components: the functional, task-solving component, the cooperative super-strate, and the communication functionality”.

By these definitions, we can sketch out the basic essences of the agent. A key property of agents is autonomy. They are independent which means they are capable of independent action without user interference. Another important property of agents is goal-driven. Agents have a purpose, and act in accordance with that purpose. Agents are also reactive. That is, an agent senses changes in its environment and responds in a timely fashion to these changes. This characteristic of agents is also at the core of delegation and automation. They are not entirely pre-programmed but can make decisions based on

information from their environment or other agents. Figure 2.1 gives an idea how an agent is situated in the environment.

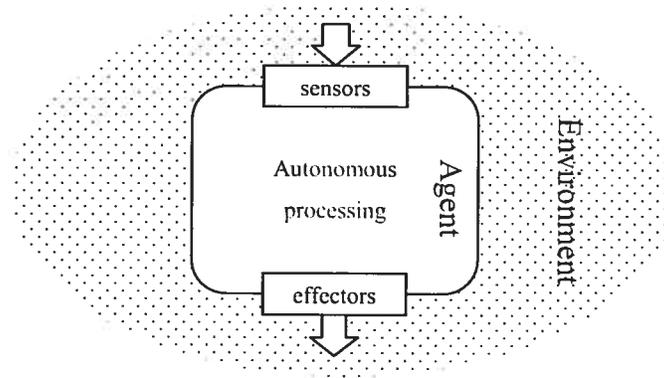


Figure 2-1: Abstract view of an agent in its environment

Chiefly, this abstract view of an agent describes that the agent receives information from its environment through different sensors and processes of these input values automatically. The agent acts in its environment, possibly changing it finally. Furthermore, the intelligent agent has a great deal in common with flexibility. Intelligent agents must be able to adapt to new environments and unpredictable situation.

2.1.2 Characteristic

The concept of agency in this work supports the weak notion of agency, and includes the common characteristics of software agents, as outlined by definitions in the table. We recognize an agent as a software module that possesses the following attributes:

- Agents are situated in and are part of some environment. They continuously exist in the environment and are continually interacting with it.
- Agents sense their environment and act upon their perceptions. An agent can detect changes in its environment and react to those in a timely matter by responding to events and initiate actions.
- Agents are (semi-)autonomous. That is, an agent has control over its own actions and is able to work and launch actions independent of other agents.

- The interaction with the environment is performed continuously and is not restricted to a specific time interval. This temporal continuity distinguishes the agent from most ordinary programs.
- Agents in a multi-agent system are communicative. An agent is able to cooperate and communicate with other agents.
- Agents are also goal-driven. An agent has a purpose and acts in accordance with that purpose. There are several ways of making goals known to an agent:
 1. A rudimentary agent could be driven by a script that predefines its actions. The script would then define the agent's goals.
 2. An agent could also be a program, as long as the program is driven by goals, and shares the other characteristics of agents.
 3. An agent could also be driven by rules, which is a more general way of defining the agent's goals.
 4. There are even more sophisticated ways of embedding agent goals, such as "planning" methodologies, and in some cases, the agent may even have the flexibility to change its own goals over time.
- Agents have to make their next behavioural decisions by consuming bounded resources, such as time and computational power. This very characteristic is called bounded rationality.

2.1.3 Classification

As with the definition of an agent, researchers have proposed a number of ways to categorize agents. In this work, we classify agents in terms of the amount of intelligence they exhibit in their behaviour. Intelligence is defined as the degree of reasoning and learning ability. At one extreme are the agents with little or no intelligence that simply react to change in the environment. They do not reason about their responses or actions, do not plan, and never learn from experience. At the other extreme are the agents that behave more like humans. They reason about their actions, make plans to achieve their

goals, and even learn from the experience and change their behaviour. This classification of agents is shown in (Figure 2-2) [Mehrdad 2003].

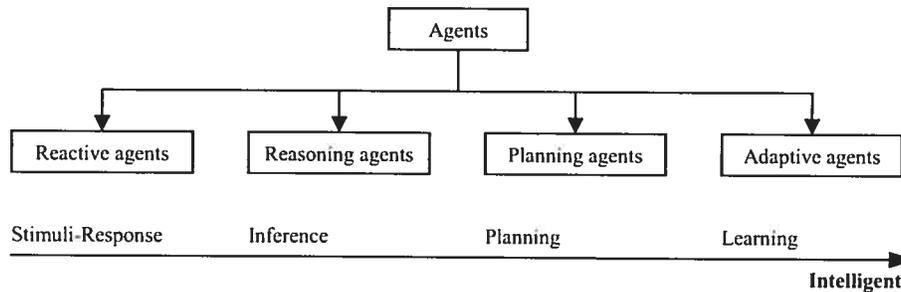


Figure 2-2: Classification of Agents

- **Reactive Agents**

Reactive or sensing-and-acting agents act or respond in a stimuli-response manner to changes in the environment. They are the simplest, fastest, and least intelligent agents and are implemented as control systems and automata. A reactive agent can be viewed as a collection of modules which operate autonomously and are responsible for specified tasks. The agent has all the knowledge that it ever needs and uses this built-in knowledge to cope with whatever stimuli it may face. Since reactive performance is computationally tractable, reactive agents are deterministic and have bounded worst-case execution times.

As a result, they are suitable for real-time applications. Moreover, many embedded applications are designed according to the principles of control theory [Rosenschein and Kaelbling 1995]. They are reactive and do not require elaborated reasoning and planning capabilities.

- **Reasoning Agents**

Reasoning agents interpret perception, use a knowledge base to draw inference, and reason to find the appropriate actions. They show a higher level of intelligence, as compared to reactive agents, by inferring and reasoning about

their responses. Reasoning algorithms are computationally expensive and have exponential complexity.

- **Planning Agents**

Planning agents, as their name suggests, compute and devise a plan (a sequence of actions) to achieve their goal. Planning is essentially automatic programming: the design of a course of action that, when executed, will result in the achievement of some desired goal. Contrary to control systems and automata, there is no well developed, generally accepted way to perform planning. The problem with planning is that it is computationally expensive. It is also hard to plan in real-time.

- **Adaptive Agents**

Adaptive or learning agents are capable of planning as well as acquiring the knowledge required for planning. The process of domain learning gives the agents the ability to do things they previously were not able to do. Adaptive agents change their behaviour based on their previous experience. Adapting requires search in domain knowledge and is computationally even more expensive than planning. Hence, it is even less tractable than planning.

2.1.4 Agent and Object

A common question that arises in the context of agent system is “how different or the same are objects and agents?” Developers use them together in the research object for most related subjects. Some consider agents to be objects, the others see agents and objects as different even though they share many things in common. In evidence, both of these two distinct notions have its own particular place in software development. We think that the agent-based way of thinking brings a useful and important perspective for system development, which is similar but still different from the object-oriented way.

As programs become more complex, the problems of local variable control and access manage turn into an issue for the modular programming. Object orientation (OO) added to the modular approach by maintaining its segments of code or methods as well as by gaining local control over the variables manipulated by its methods. In OO, objects are considered passive because their methods are invoked only when some external entity sends them a message. In addition, the data-abstraction is achieved by users defining their own data-structures - objects. These objects encapsulate data and methods for operating on that data. Furthermore, the OO allows new objects to be created that inherit the properties (both data and methods) of existing objects. This allows archetypal objects to be defined and then extended by different purposes, which needn't have complete understanding of exactly how the underlying objects are implemented.

In contrast, software agents have their own thread of control, localizing not only code and state but their invocation as well. Such agents can also have individual rules and goals, making them appear like "active objects with initiative." In other words, when and how an agent acts is determined by the agent [Odell 2002].

Agents are commonly regarded as autonomous entities, because they can monitor their environment for the own set of their internal responsibilities. Furthermore, agents are interactive entities that are capable of using the external messages. These messages can support method invocation as well as informing the agents of particular events, asking something of the agent, or receiving a response to an earlier query. Lastly, because agents are autonomous they can initiate interaction and respond to a message in the way they choose. Instead of physically launching an agent method, agents can inspire themselves because of the interactive and autonomous nature. Van Parunak in [Parunak and Van Dyke 1997] summarizes it well: "In the ultimate agent vision, the application developer simply identifies the agents desired in the final application, and the agents organize

themselves to perform the required functionality.” No centralized thread or top-down organization is necessary since agent systems can organize themselves.

While we can develop an agent architecture using an object-oriented framework, the OO approach also has some kinds of the attributes about the behavioural autonomy of the agents, i.e. the ability of accessing their methods is controlled by the object itself. The process of hiding data and associated methods from other objects is achieved by specifying access permissions on object-internal data elements and methods. By offering functionality through the public methods, object internal data is invisible from outside the object. The locus of control is placed upon external entities that manipulate the object through its public methods [Joseph and Kawamura 2001].

In the agent approach, we could think about objects as agents that make requests of each other. Since the agent’s actions are voluntary as opposed to be invoked by the caller in an OO environment, agent systems would provide more fine-grained access and security control through an agent communication language interface.

Thus, the important aspect of the Agent Oriented approach is that, in opposition to object method specification, an agent communication language interface requires that the communicating parties must be declared allowing the agent to control access to its internal methods, and thus its behaviour. This in itself means that the agent’s objectives must be considered, even if only in terms of which other entities the agent will collaborate with [Joseph and Kawamura 2001]. Table 2-2 summarizes the evolution of programming languages.

Table 2-2: Evolution of programming approaches

	Machine Language	Structured Programming	Object Oriented Programming	Agent Oriented Programming
Relation to Previous level		Bounded unit of program	Subroutine + Persistent local state	Object + Independent thread + Initiative
Structural Unit	Entire program	Subroutine	Object	Agent

	Machine Language	Structured Programming	Object Oriented Programming	Agent Oriented Programming
How does a unit behave? (code)	External	Local	Local	Local
What does a unit do when it runs? (state)	External	External	Local	Local
When does a unit run?	External called	External called	External called	Local (rules; goals)

Originally, the basic unit of software was the complete program where the programmer had full control. The program's state was the responsibility of the programmer and its invocation determined by the system operator. The term modular did not apply because the behaviour could not be invoked as a reusable unit in a variety of circumstances.

As programs became more complex and memory space became larger, programmers needed to introduce some degree of organization to their code. The modular programming approach employed smaller units of code that could be reused under a variety of situations. Here, structured loops and subroutines were designed to have a high degree of local integrity. While each subroutine's code was encapsulated, its state was determined by externally supplied arguments and it gained control only when invoked externally by a CALL statement. This was the era of procedures as the primary unit of decomposition.

In contrast, object orientation added to the modular approach by maintaining its segments of code (or methods) as well as by gaining local control over the variables manipulated by its methods. However in traditional OO, objects are considered passive because their methods are invoked only when some external entity sends them a message.

Software agents have their own thread of control, localizing not only code and state but their invocation as well. Such agents can also have individual rules and goals, making them appear like active objects with initiative. In other words, when and how an agent acts is determined by the agent.

2.2 Online learning system

The online learning systems are a class of important services in which the information infrastructures provide for learners through the Internet. They base their operation on access to information, electronic transaction and communication services provided by the information infrastructures. Regarding the teaching process, new paradigms enable distribution of educational content, interaction among classes of learners (instructor-learner, learner-learner and learner-educational institution), testing, evaluation and eventually advice in different domain knowledge areas.

2.2.1 Characteristic

Development of information technology enabled foundation of the online learning systems that possess the following characteristics [Rosić *et al.* 2002]:

- Distribution

Knowledge to which the online learning systems enable access can't be found in one place but is distributed at more places which enable faster work of these systems along with making access to the systems' resources possible for a large number of learners. In other words, distributed knowledge is clustered into a simple knowledge-base through the online learning system.

- Adaptability

The systems have the possibility of adaptation to the learner's habits and needs. Because of the variety of the online learners, an online learning system has to consider the flexibility and adaptability of the system.

- Multimedia orientation

The learning systems are based on multimedia presentations of domain knowledge to the learners. Internet-based online learning system provides the possibility to represent the multi type's information over the network which does

not just include the textual documents and the image, but also the audio and video, even including the online radio and television.

- Cooperation

Different systems support common elements which are available to all learners (for example, different learning systems can use common databases with domain knowledge. Reference can cross all over the Internet).

It is expected that online learning systems will in the near future keep all these characteristics along with additional development of adaptability of the system to learners as well as further development of cooperation. Online learning systems will have a dynamic life cycle where there will not be a clear line between the phase of development and usage of the system. The online learning system itself and its environment will be frequently updated in response to changes in: information, services, hardware, software and learners' requirements. The systems will enable diversity in information formats and execution platforms. Furthermore, learners of online learning systems will range from expert to novice with a wide variety of purposes. It is necessary to provide the use of services for all those groups in a qualitative way. No matter to which group the learner belongs, increase of quantity of the available information and the services offered via the information infrastructure often puts the learner in a state of information overload, where the learner's productivity decreases due to processing too large amount of received data.

2.2.2 Agent-based online learning system

In the online learning system, agents could be a cognitive user tools belong to system and providing essential help when the user requested them. The two major roles of an agent in the online learning system are [Baylor 1999]: 1) cognitive tools 2) intelligent tutors. According to achieve functions of the two roles above, agent should achieve four responsibilities following [Huang and Edwards 2003]:

- Helping learner effective to find his/her personal learning styles.
- Navigating learner to motivate him/her.
- Watching and evaluating learner's attitude anytime.
- Organizing information and course for learner.

During the learning session, agent should support, guide, and extend the thinking processes of their students. In terms of educational psychological theory, the concept of distributed cognitions readily applies to intelligent agents since they could be used to serve as extensions of a person's intellectual capacity. By extending the cognitive capabilities of the learner, intelligent agents could decrease the learner's limit of his ability to imitate processes demonstrated by others.

To best improve the learning process, agents actively participate in the learning activities rather than passively retrieve information. In this way the agent provides an environment where the learner has a personal learning secretary and/or partner whom makes him think harder and more deeply about the content, and using the agent as a natural cognitive extension. In this manner, the agent also could serve as a teacher, prompting the learner to engage in analysis of his/her own cognitive processes, and promoting the learner to consider what strategies are being used through the learning session.

2.2.3 Multi-agents system

Sometimes the problems in an agent-oriented system are too large and complex for a centralized agent to handle. Sometimes we have to interconnect or interoperate with the legacy system in the agent system. Sometimes the problems and recourses are inherently distributed. In all these situations, we must consider a multi-agent's learning system which contains a number of agents which interact with one another through communication. The agents are able to act in an environment; where each agent will act

upon or influence different parts of the environment. The motivation of a multi-agent system including [Pinar 1996]:

- Solve problems those are too large for a centralized agent
- Allow interconnection and interoperation of multiple legacy systems
- Provide a solution to inherently distributed problems
- Provide solutions which draw from distributed information sources
- Provide solutions where expertise is distributed
- Offer conceptual clarity and simplicity of design

It is possible to organize the execution of agents in the environment of a multi-agent system. The multi-agent systems have the following characteristics:

1. Each agent may have different knowledge, capabilities, reliability, resources, responsibilities or authority.
2. Different agents may perceive the same event or object differently.
3. The agents may specialize in or focus on different problems and sub-problems.
4. An important goal is the convergence of solutions despite the incomplete or inconsistent knowledge or data.

It is clear from the mentioned characteristics that the realization of the multi-agent system is complex because with such systems the global supervision of the system most often doesn't exist, each and every agent has the access to the limited set of data, and each and every agent isn't able to independently solve the set problem. The data is distributed and the agents have to communicate with each other while solving the set problem.

In the multi-agent system more personal agents that cooperate with each other are assigned to the user. In addition, in such environments the agents assigned to different users also cooperate (see Figure below).

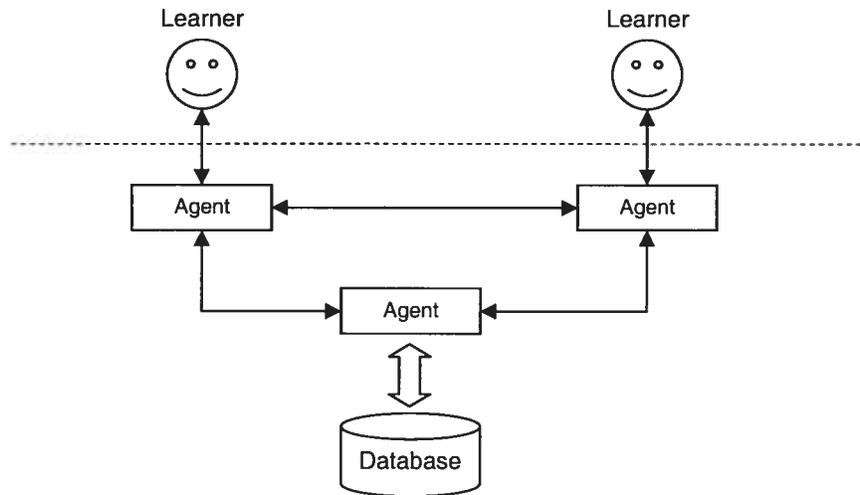


Figure 2-3: Multi-agent system

We can even say the communication and cooperation between the agents are the essential features of multi-agent system. They negotiate and discuss one another. Agents are organized into team formation to support a collective goal. Information is shared among team members. They joint beliefs, goals and plans. Some of the benefits of using multi-agent system including:

- Speedup and efficiency – due to the asynchronization and parallel computation
- Robustness of reliability – “graceful degradation” when an agent fails
- Scalability and flexibility – easy to add new agents
- Cost – assumption - less communication cost since less need to transform raw data
- Development and reusability – easier to develop and maintain a modular software

2.3 Online learning system model

In the National Academy of Sciences supported book, “How People Learn” [Bransford, *et al* 2002], John Bransford and a distinguished group of scholars reflect on the question of what constitutes a good learning environment. They suggest the good learning

environments are characterized by these areas of emphasis. They are learner centered, knowledge centered, assessment centered and community centered.

- **Knowledge Centered** - Outcomes oriented - knowledge, skills, and attitudes needed for successful transfer.
- **Learner Centered** - connect to the strengths, interests, and preconceptions of learners and help them learn about themselves as learners.
- **Assessment Centered** - provide multiple opportunities to make students' thinking visible so they can receive feedback and be given a chance to revise.
- **Community Centered** - environment where students feel safe to ask questions, learn to work collaboratively, and are helped to develop lifelong learning skills.

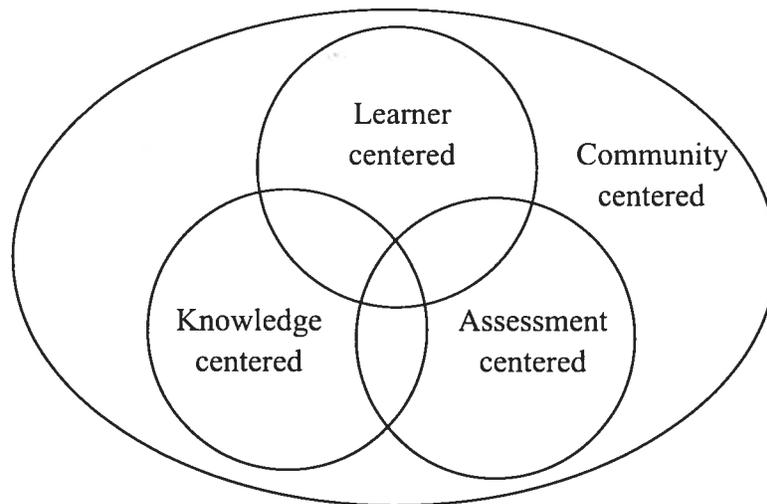


Figure 2-4: A Developing Conceptual Framework for Online Learning

Knowledge Centered Learning Environments are carefully based on what we want learners to know and be able to do when they finish with our materials or course and provide them with the foundational knowledge, skills, and attitudes needed for successful transfer.

Learner Centered Environments connect to the strengths, interests, and preconceptions of learners and help them learn about themselves as learners.

Community Centered Learning Environments provide an atmosphere both within and outside the classroom – where students feel safe to ask questions, learn to use technology to access resources and work collaboratively, and where they are helped to develop lifelong learning skills.

Assessment Centered Learning Environments provide multiple opportunities to make learners' thinking visible so they can receive feedback and be given a chance to revise their mental models.

2.4 Summary

This chapter started out by explaining agent's definition and its major properties, and then we also described classification of agent. After that, we present a comparison of the agent and object oriented system. And then, we introduce the online learning system and its characteristic. We also present the agent based learning system. At the end of this chapter, we describe an online learning model.

Personalized learning support

Throughout the chapter, we will be referred to a number of background references for more details about the personalized learning support (PLS).

3.1 Personalized learning support

Personalized learning support analyse the learning style, the learning process and the learning results of the learner to adjust the curriculum of the learning system according to the learner's knowledge level, adapt the selection of learning material like presentations, examples, illustrations, feedback, tests etc.

Personalized learning support (PLS) focuses on the dominant factors that impact self-motivation, self-directedness, and learning autonomy. It is based on research into the neurobiology of learning and memory, and incorporates the dominant impact of emotions, intentions, and social factors, as well as cognitive issues. PLS explores design of the online learning environment, online presentation of instruction, the role of the instructor, and expected outcomes. It also describes strategies to help learners improve online learning ability as they become more self-motivated, self-managed, independent learners.

3.2 Comparison

In this session, we present a simple comparison between the traditional approach and personalized learning support (PLS).

Focus of design

In traditional approach, the learning topic is limited by the mentor’s ability. Design often centers on what he/she likes to do or is comfortable to do. On the contrary, personalized learning topic design centers on what and how the learner will learn on their own. It provides a learner centered learning environment to the learner, instead of knowledge centered or assessment centered.

Traditional	PLS
<ul style="list-style-type: none"> • what likes to do or is comfortable to do 	<ul style="list-style-type: none"> • what and how the learner will learn on their own

Learning process (What will learners learn?)

Traditional learning processes are normally fixed by the materials. Conversely, PLS process is based on the learner’s completion behaviour, which is adjustable and dynamic.

Traditional	PLS
<ul style="list-style-type: none"> • Based on Textbooks • Centers around chapters • Focus on covering the material 	<ul style="list-style-type: none"> • Based on task analysis and needs assessment • Emphasized application of knowledge, skills, abilities • Focus on what learners will be capable of upon successful completion

Progress evaluation (When have learners learned?)

Traditional approach’s progress evaluation is based on the exam which is a norm-based assessment. In contrast, PLS achievement measures against pre-stated performance standards. Learners only progress when competency is mastered.

Traditional	PLS
<ul style="list-style-type: none"> • When an exam is passed • Completion often based on “seat time” 	<ul style="list-style-type: none"> • Relies on performance demonstration of skill, knowledge, and attitudes • Achievement measured against

<ul style="list-style-type: none"> • Norm-based assessment 	pre-stated performance standards <ul style="list-style-type: none"> • Learners only progress when competency is mastered
---	--

Achievement (How will learners develop)

Because learners have different roles in traditional learning approach and PLS, passive in the former and active in the latter, learners' achievements are also diverse.

Traditional	PLS
<ul style="list-style-type: none"> • Lecture-based: relies on faculty to deliver instruction • Places learner in passive role • Often offers little variety in learning style • Connections between intended outcomes and learning activities blurred 	<ul style="list-style-type: none"> • Features learner-centered activities • Places learner in active role • Offers varied learning activities for varied learning styles • Ties learning activities to intended outcomes

3.3 Challenges

The transition of instruction from classroom teacher-directed to online user-directed has not always been smooth. In the traditional classroom, students learned to depend (often too much) on instructors for their motivation, direction, goal setting, progress monitoring, self-assessment, and achievement.

In contrast, online learners find that they need to take greater responsibility for their own learning. But too many are unprepared or unwilling to do so. Perhaps this is because as adult learners their learning skills are the product of years of development and habituation in the classroom.

In an online learning environment, it's clear that most learners have had little time to acquire and perfect a satisfactory online learning ability. It is not surprising that completion rates in online courses are low, since the majority of today's solutions rely on traditional classroom design perspectives. Satisfying learners online and ensuring that they will capably finish courses, achieve objectives, and acquire new knowledge and skills are today's online learning challenge.

Meeting this challenge requires a better understanding of the psychological sources that influence an individual's online learning ability and how a learner may want or intend to learn. Specifically, the search for more sophisticated learning theories requires a better understanding of online learning process.

In the past, explanations of differences in the ways that people learn have focused on cognitive factors having to do with thinking and information processing, such as learning styles. However, now several areas of research point towards the important effect of emotions on successful personalized learning. This kind of research supports better designs because it addresses a more comprehensive set of key psychological factors. Consequently, we can determine the issues that may particularly frustrate or encourage the learning audience.

However, information processing, knowledge and skill building are still important considerations in the design of instruction. These primarily cognitive aspects support more traditional approaches. Learners who are more dependent on the teacher-pupil relationship benefit from having an instructor to promote learning and manage needs (e.g., emotions, intentions and social issues). Hence, it is important for the learning process both to address fundamental learning needs and to specifically promote self-directed and self-motivated learning.

3.4 The Online Learning Ability

Successful instructors and trainers know that they can make a huge difference in the classroom with personalized attention, particularly in recognizing and tapping into how individuals may need to learn differently. As good instructors, they intuitively manage key human factors (e.g., passion, happiness, dislike, fear, striving, will, frustration, satisfaction, and anger) to promote learning. Online these factors may be overlooked.

By considering the impact of emotions and intentions, we can better understand how and why individuals learn differently. For example, some learners are happiest learning in collaborative, facilitated environments with learning tasks accomplished in a structured or linear fashion. Other learners thrive in competitive learning environments that focus on specific details, tasks, and projects. Some learners are passionate about exploring new challenges and taking risks, and they enjoy using learning to achieve long-term personal goals. Finally, some learners are formally or situational resistant to any kind of learning that appears to have little value or benefit to them.

Some online learning models consider these important distinctions between learning types and, when necessary, try to manage these differences. Translating this kind of psychological information into learning strategies helps designers create learning situations that work best for the intended audience. As we put more learners online, we expect them to take on more responsibility for their learning, raise their online learning achievement, and improve their ability over time. We will begin to see how each person may or may not need additional or reduced support. At the same time, key success attributes and patterns will emerge that identify gaps in people's readiness to engage in online learning.

Understanding learning differences allows us to tap into key psychological factors that will help people to learn online successfully.

Personalized learning portrays characteristics, influences, and relationships between three key construct factors: (1) conative (emotions) and affective (intentions) intrinsic motivational aspects, (2) self-directed strategic planning and committed learning effort, and (3) learning autonomy. Combined, these three factors greatly influence an individual's general approach to learning. This model offers explanations for fundamental learning differences, and suggests specific strategies for accommodating learning needs for audiences differentiated by learning type.

Learning orientations are an effective way to segment the audience according to higher-order psychological factors (e.g., affective, conative, and social outlooks). These factors foster how we develop, manage, and sometimes override our cognitive learning preferences, strategies, and skills.

Profiles or archetypes have been developed for these orientations to describe their emotions and intentions with respect to learning and performance. These profiles provide specific scales for measuring common learner-difference attributes (e.g., high-to-low motivation, self-directedness, and autonomy). The learner-difference profiles can also guide analysis and design of instruction and environment. The result is a set of tailored solutions that help raise learning ability and that improve the learning experience.

3.5 PLS design

As designers, we can collect and analyze information about how individuals learn in a given situation and more effectively provide personalized solutions.

Collecting critical success attributes common to the learning group is vital in helping learners improve learning ability, understand how they learn best, and make educated choices about managing their learning environments.

- Shifts more responsibility to learner
Since learner has become the center of learning process, learner's responsibility should be also emphasized to enhance the motivation, self-directedness and autonomy.
- Reduces inefficiencies in learning process
In a self-prompting PLS environment, it's the duty of designer to provide an efficient learning process to learner.
- Changes focus of learning from: process to outcomes
Emphasizing the learning outcomes instead of process, that is one of principal strategies in PLS.

Personalization includes using learner-specific strategies that may take many forms as it adapts environments and offers alternative choices, including sequencing or presentation of content, practice, feedback, and assessment. Good instructors have been offering these personalization strategies in classrooms for years. In online learning situations, technology should ensure that these same strategies can be applied and increasingly self-managed by the online learners over time.

Basing instructional analysis, interpretations, and decisions on a standardized multidimensional framework developed by identifying critical success attributes helps to formalize the personalization process. Once organized for the targeted audience, the framework can be used to create a blueprint for more personalized learning.

The blueprint for personalized learning should use well-developed criteria based on iterative cycles of measurement to track each learner's interaction with the personalized solution. Results should measurably show how the learning solution becomes more valuable to the learner. The desired result should show an increased loyalty and affinity for the online learning solution over others.

The goal for personalized learning: Two individuals accessing the same personalized instruction simultaneously may see different presentations and progress and improve differently-- with greater satisfaction.

3.6 Personalization Framework

There are many ways to personalize learning. Using a well-tested personalization framework helps ensure that solutions and interpretations are consistent, relevant, and useful with measurable improvements. The personalization framework described here has four levels or perspectives. The fourth level has five major dimensions. From the simplest to the most complex, the dimensions for them are: 1) name recognition; 2) self-managed; 3) segmented; 4) cognitive-based; and (5) whole-person-based. Each dimension has a specific purpose and resulting impact. Your targeted goals and outcomes should govern your choice of these dimensions. These dimensions can work separately or in tandem to enhance the personalized learning experience.

- **Name Recognition Personalization**

Name recognition personalization is useful because most people value being acknowledged as an individual. As an example, the learner's name appears at the top of the screen or previous accomplishments are marked.

- **Self-Managed Personalization**

Self-managed personalization enables learners (using questionnaires, surveys, registration forms, and comments) to describe preferences and common attributes. As an example, learners may take a pre-course quiz to identify existing skills, learning preferences, or past experiences. Afterwards, solutions appear based on the learner-provided answers.

- **Segmented Personalization**

Segmented personalization uses demographics, geographics, psychographics, or other information to divide or segment learning populations into smaller, identifiable and manageable groups for personalization. As an example, learners that share a common job title or work in a certain department would receive content based on prescriptive rules that would support the learning and performance requirements for that specific segmented group.

- **Cognitive-Based Personalization**

Cognitive-based personalization uses information about learning preferences or styles from a primarily cognitive perspective to deliver content specifically targeted to differing learner attributes. As an example, learners may choose to use an audio option because they prefer hearing text rather than reading it. Or, a learner may prefer the presentation of content in a linear fashion, rather than a random presentation with hyperlinks. This type of personalization operates on more complex algorithms than the other types and is able to factor more learner attributes into each interaction. This type of personalization generally works by collecting data, monitoring learning activity, comparing that activity with other learner behaviour, and predicting what the user would like to do or see next.

- **Whole-Person Personalization**

Whole-person personalization seeks to understand the deep-seated psychological sources (more than the conventional cognitive-based prescriptions) impacting differences in learning behaviour, make predictions about delivering content, and deliver content specifically to help the learner achieve learning objectives and more importantly, improve learning ability and enhance online learning relationships. As the individual learns, the system also learns as it collects data, tracks progress, and compares responses and common patterns to improve responses, i.e., it becomes more precise over time. In its most sophisticated form, whole-person personalization requires real-time personalization to modify responses to a learner based on a changing perception throughout the learning experiences, as it occurs (like an instructor in the box).

3.7 Summary

In this chapter, we present the PLS model and its characteristics. Personalized learning support has become one of best solution for the online individual learning environment. After present a comparison between traditional learning and PLS, we also discuss the challenge of PLS and its ability.

Cooperative learning environment

Consider a multi-agent learning system, where new learner enters the system already populated with experienced learners. The new learner begins with a blank slate, as he has not yet had an opportunity to learn about the learning environment (although the agent of the new learner may of course be “hard-wired” with behaviours that will probably turn out to be useful). However, the agent may not need to find out everything about the environment for itself: it may well be possible to benefit from the accumulated learning of the population of more experienced agents. This situation could describe highly autonomous software agents operating in a learning system.

In recent years there has been some progress towards understanding the adaptive value of cooperative learning [Davis and Sklar 2003] [Lesser 1999] [Plaza and Ontañon 2003]. Some of the conclusions are rather straightforward: cooperative learning is more likely to evolve when the costs of individual learning are high [Kameda and Nakanishi 2003]. In a cooperative learning environment, the signification for a software agent might be a situation where mistakes are financially costly for the agent’s owner.

In this chapter we propose our vision of the cooperative learning environment (CLE) based on the interaction between human and human, human and agents, agent and agent. We give first a brief study of the CLE, and then we present the environment, define the conceptual model of the system and our multi-agent architecture respectively.

4.1 Cooperative learning

Cooperative learning is a teaching strategy in which small teams, each with learners of different levels of ability, use a variety of learning activities to improve their understanding of a subject. Each member of a team is responsible not only for learning what is taught but also for helping team-mates learn, thus creating an atmosphere of achievement. Learners work through the assignment until all group members successfully understand and complete it.

4.1.1 Benefit

Through active participation and more on-task behaviour, learners will benefit from higher academic achievement for all. Benefits include improved social skills, higher self esteem, greater use of higher-level thinking skills, and increased appreciation for different points of view.

- promote learner learning and academic achievement
- increase learner retention
- enhance learner satisfaction with their learning experience
- help learners develop skills in communication
- develop learners' social skills
- promote learner self-esteem
- help to promote positive race relations

4.1.2 Elements of Cooperative Learning

Cooperative learning (CL) is instruction that involves learners working in teams to accomplish a common goal, under conditions that include the following elements:

1. Positive Interdependence

- Each group member's efforts are required and indispensable for group success

- Each group member has a unique contribution to make to the joint effort because of his or her resources and/or role and task responsibilities

2. Face-to-Face Interaction (promote each other's success)

- Explaining how to solve problems
- Teaching one's knowledge to other
- Checking for understanding
- Discussing concepts being learned
- Connecting present with past learning

3. Individual & Group Accountability

- Keeping the size of the group small. The smaller the size of the group, the greater the individual accountability may be.
- Giving an individual test to each learner.
- Observing each group and recording the frequency with which each member-contributes to the group's work.
- Assigning one learner in each group the role of checker. The checker asks other group members to explain the reasoning and rationale underlying group answers.
- Having learners teach what they learned to someone else.

4. Interpersonal & Small-Group Skills

Social skills includes: leadership, decision-making, trust-building, communication, and conflict-management skills.

5. Group Processing

- Group members discuss how well they are achieving their goals and maintaining effective working relationships
- Describe what member actions are helpful and not helpful
- Make decisions about what behaviours to continue or change

4.2 Online cooperative learning

Cooperative learning takes advantage of learning as a social process. Learners are frequently more motivated to work, when there is an audience beyond that of the teacher. Additional benefits of cooperative learning are many. Learners can access interesting source data, experience virtual travel, and connect with other learners and subject experts to study and learn together.

Learners frequently improve their reading, writing, and data management skills. Online Cooperative learning provides effective opportunities for learners to practice learning new languages, by connecting non-native speakers with native speakers. Web-based collaboration often presents a positive public forum for showcasing learner work for parents and the community.

Geography, history, politics and world cultures become more relevant to learners as they communicate directly with other learners from distant locations. Learning is more meaningful when, for example, learners who are studying volcanoes can communicate directly with children living at the foot of Mount Kilauea in Hawaii and learn first hand about flowing lava, spewing ashes, and seismic activity. When they can see how the subject matter affects their everyday lives, they're eager to contribute. In Southern California and Kobe, Japan, middle learners learn about earthquake preparedness by sharing experiences through Internet video conferencing. High school learners receive first-hand accounts of life in a besieged Bosnian town and anxiously hope for resolution. Elementary school Learners raise awareness about environmental issues by tracking key data on an international scale.

Newsday is an example of an interdisciplinary project that results in a newspaper produced by learners. Working as reporters, learners throughout the world submit feature

articles to the Newsday news wire. Team members collaborate and work as editors, graphic artists, and publishers to produce the paper together. Since the newspaper is global, learners also gain a broad knowledge and understanding of current events and international issues.

Community-oriented learning can mobilize the energy, commitment, and idealism of young people, while teaching them leadership skills and personal responsibility. These “service-learning” cooperative projects provide an opportunity for learners to apply newly learned skills to real world situations, thus increasing retention. And, reciprocally, the community benefits from cooperative projects that increase pride in the community, help reduce vandalism, discourage graffiti, bullying, and school violence, or provide food, clothing and assistance for the needy, sick and elderly.

Learners in a cooperative learning environment are active learners, who construct knowledge, rather than passively absorb it. Effective collaboration requires coordinated scheduling, common communication tools and mutually accepted goals and objectives. Well-designed online cooperative learning projects provide learners with unique and highly motivating learning experiences that would not be available to them within the traditional classroom walls.

4.3 Existing online CLE

Cooperative learning has been around for a long time [Dumas 2003]. However the use of computer to support such activity is fairly new. Computer Supported Cooperative Learning is a new emerging paradigm that extends classical Intelligent Tutoring System by introducing the concept of cooperation. In this sense, some researchers proposed the learning with companion approach [Chan 1990], which simulates a second learner who learns together with the learner. Another proposed the learning by teaching model where

the learner could teach the learning companion by giving explanation [Palthepeu 1991], etc. The follows sections present some of the existing cooperative learning models.

4.3.1 Simulated student agent

Aurora Vizcaino, Benedict du Boulay present a simulated student model in their paper “Using a Simulated Student to Repair Difficulties in Collaborative Learning” [Vizcaino and Boulay 2002]. In that model, a simulated student is used to repair the learner’s difficulties in the cooperative learning environment. When the learner is too passive or when he leaves the learning topic for a long time. A simulated student will post message in the shared chat window to intervene the student’s learning process.

The problem situations are grouped into 3 categories:

- Problem solving
Including the learner doesn’t know how to work, posts wrong solutions, or has different point of view about the solutions, etc.
- Off-topic conversation
That’s the situation when learners talk about other topic for a long time.
- Passive students
Including the students who have deficient knowledge, have adequate knowledge, or the hyperactive students

The agent monitors the learners’ practice during the learning session. According to the problem situations above, it tries to detect the learners’ difficulties and act as a real student in the CLE to help them work out the difficulties by joining their conversations.

4.3.2 Computational model of distance learning

This cooperative learning model is presented in the paper “A Computational Model of Distance Learning Based on Vygotsky’s Socio-Cultural Approach” [Andrade *et al.* 2001].

In this model, the cooperative learning environment privileges collaboration as form of social interaction. Four agents are included in the system to support the cooperative learning. All the individuals in the system are described as integrated social agent.

ZPD agent and mediating agent monitor the learner's behaviour to provide personal learning support in the environment. ZPD (Zone of Proximal Development) - a pedagogical statement defined by Vygotsky is *“the distance between the real level of development, determined by the capacity to solve a problem independently, and the level of potential development, determined by the resolution of a problem under an adult's orientation or in collaboration with other more capable students”*.

Social agent and semiotic agent provide the social support to the system. The former establishes the integration of the society and to construct the student group. The latter assists the cognitive activity by introducing the external stimulation.

In this model, ZPD agents interact with the social agent in the search for partners to assist the learner in the learning process. The mediating agents interact with the semiotic agent to obtain the symbols that should be presented to the learners. Following figure (Figure 4-1) is an architecture sketch of the social learning model.

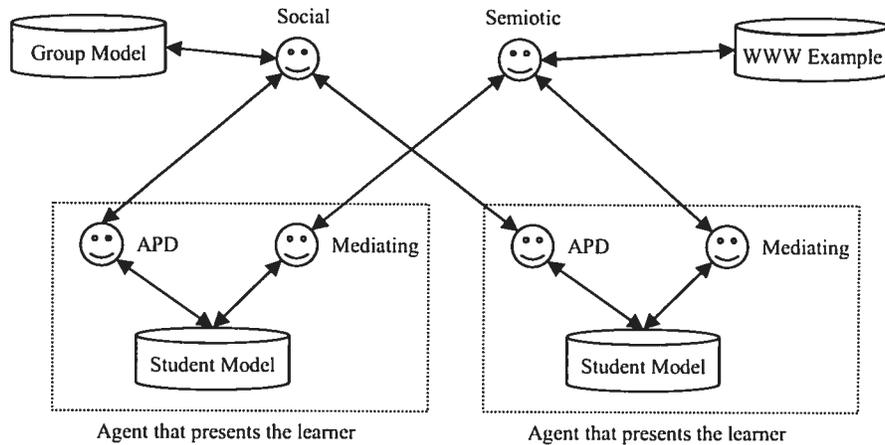


Figure 4-1: Social learning model

4.3.3 Multi-agent architecture for cooperative learning

Thierry Mengelle and Claude Frasson present a multi-agent architecture of ITS in their paper “A multi-agent architecture for an ITS with multiple strategies” [Mengell and Frasson 1996] which focus on the characteristic of “learning by disturbing” in the CLE.

The architecture is structured to several distributed pedagogical agents which are considered to be the actors assumed the pedagogical roles. Since the pedagogical agents are assigned by the specific pedagogical strategy of the learner, the communication between the learner and the system becomes more flexible. For instance, the peer-to-peer tutoring strategy involves two agents only – the tutor and the artificial learner agent. On a different scene, the same agent can play different roles in accordance to the chosen strategy. The paper’s authors used a new statement – “actor” to define this kind of pedagogical agents.

In the "Learning by disturbing" strategy, the tutor agent could request the troublemaker agent to intervene the learner by giving the incorrect solutions which aims to strength the learner’s self-confidence. Finally, according to the answer of the learner, the tutor approves or congratulates him, or gives him the right solution.

4.4 Agents in cooperative learning

The essential feature of cooperative learning is that the success of one learner helps other learners to be successful. In CLE, learners work together to enhance their own and others practice by exchanging their learning experiences. The goal is reached through interdependence among all PLEs in the environment rather than working alone.

Cooperative learning is important because it produces greater learner achievement than individual learning methodologies. Beyond the academic benefits, we distinguish the social benefits (development of the learner social skill), the economic benefits (less time and material are needed), etc [Labidi and Silva 2000].

As it is shown, education is fundamentally a cooperative process. An important issue is to study how this cooperation could be supported? One of the methods is that cooperation could be supported exchanging actions. All the agents interact by monitoring the others' action, which could be considered as mediating tools that support exchanges of viewpoints and concepts between the learners.

Cooperative learning system using agent always has a number of agents in it, but multi-agent system is not always a cooperative system. An important difference between the multi-agent system and cooperative agent system is that the policies/goal in the multi-agent system are fixed on each agent and usually they are different, which means agents cannot improve its rewards by changing its policy even they are in a same environment. In contrast, cooperative agents use the same reward functions for all the agents. Therefore, optimal policies can be represented on all the participant agents.

On the other hand, cooperative agents system cannot be seen as a single agent simplistically although they are in a same environment and use the same reward functions. In a single agent learning environment, there's only one agent that decides which policy to use. In contrast, the system behaviour is influenced by all member agents in a CLE. Agents do not only coordinate their behaviour but refine the other agents' behaviours with its own experience and try to detect the information about its potential action capabilities.

4.5 Mechanisms

Turning to the question of mechanism: there are many ways in which one agent might learn from the behaviour of another. In the cooperative learning, there has long been a focus on imitation, i.e., the goal-directed copying of another's behaviour. However, as some researchers point out, true imitation is a complex process that seems to involve not only perceiving and reproducing the bodily movements of another, but understanding the changes in the environment caused by the other's behaviour, and finally being able to grasp the "intentional relations" between these, i.e., knowing how and why the behaviour is supposed to bring about the goal. Much of the work on imitation has been short on specifics about the underlying mechanisms [Alonso *et al.* 2001].

We will instead consider a range of simpler mechanisms that could easily be implemented in software agents. It has long been recognized within fields like artificial life that complex global phenomena can arise from simple local rules, and this is precisely what is happening in many cooperative learning contexts: individuals follow a simple rule and, in combination with some form of learning, this gives rise to an apparently sophisticated cooperative learning system at the group level. From the point of view of building learning abilities into intelligent agents, simple mechanisms have obvious advantages in terms of robustness and design costs.

Communicable behaviour is exemplified by a rule such as “If others are unsatisfied with the answer, it must be a discontented answer.” The idea is that the stimulus produced by the performance of a particular behaviour serves as triggers for others to behave in the same way. Note that this does not involve real learning, and is merely a reactive system, but could nevertheless produce adaptive social behaviour.

Stimulus enhancement (also called local enhancement) is what happens when agent obey a rule like “Follow someone older than you, and then learn from whatever happens.” A simple behavioural tendency combines with the capacity for learning to result in the potential transmission of acquired behaviours.

Observational learning If we add slightly more sophisticated learning abilities to stimulus enhancement, we get observational learning. The algorithm involved is approximately “Pay attention to what others are doing or experiencing, and if the results for them appear to be good or bad then learn from this.” Observational learning can also exist in a simpler form: explicit evaluation of the others experience as good or bad may be omitted.

Matched-dependent behaviour Species such as simple reinforcement learning can result in cooperative learning if the contingencies are right. There is no implication that the follower understands the leader’s intentions, nor even that the follower is aware of the match between the leader’s behaviour and its own. The general point is that contagious behaviour may sometimes be learned.

Cross-modal matching Vocal mimicry by birds is often held to be a special case of social learning: because the original stimulus and the animal’s response are in the same

sensory modality, a relatively simple pattern-matching mechanism could account for the phenomenon. In contrast, copying the movements of another animal requires cross-modal matching; the observer must be able to translate the visual input associated with another's movements into appropriate motor outputs. Consider that there is no trivial link between the sights of watching someone else scratches their nose, and the experience of scratching your own nose.

4.6 PLE & CLE

In our cooperative learning environment (CLE), a set of agents constitute a personal learning environment (PLE) to provide the personalized learning support. The PLE presents the individual learner in the CLE learning group. Learners communicate each other inside the CLE group. As illustrated in the following figure (Figure 4-2), the PLE is modeled as the interaction of several heterogeneous agents. We have identified:

1. Personal agent: learner's profile manager and search agent;
2. Problem agent: learner's problem detector;
3. Motivation agent: learner's motivation detector.

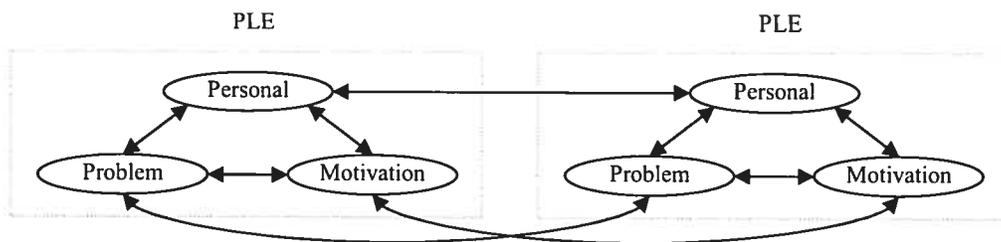


Figure 4-2: Communication in CLE learning group

Personal agent has the role of presenting the knowledge to the different learners in the different cooperative areas. It's also responsible for exchanging the learners' comments and recommendations during the learning practice. It interacts with the problem agent

and motivation agent for selecting the adequate cooperative learning topic to be applied. The personal agent is also responsible for grasping the learners' unanswered question into a pending database which could be answered by the other agents in the future.

Problem agent defines and proposes the learning topic to the learner. It tries to find out the learner's weakness and missing knowledge by analyzing the learning history and comparing it with other learners in the CLE learning group. According to this analysis, it proposes an appropriate learning topic to the personal agent and pre-fetches the necessary information and related documents (e.g. reference from the Google search engine, topics in the same category, topics associated from other topic and other learners' recommendation/commentary about the current topic etc.).

Motivation agent also watches the learner's behaviours, but it focus on the learner's motivation detection. Sometimes, a new learner could be difficult to start a learning process when he faces a new environment. He does not know where to search, what to look at, etc. The same difficulty could frequently happen when the learner transfer to a new subject. Motivation agent will try to detect the learner's motivation in this situation and propose some suggestions to the learner through the personal agent by using the presentation or dialogue.

On the other hand, when learners can obtain related information and knowledge by visiting the linked web pages, the learner may lose his way in the linked web pages, strays from the proper learning topic for a long time. The agent also monitors the learner's navigation history to detect this kind of problem and prompt the learner to return to the learning topic.

Personal, problem and motivation agents work together In PLE to provide the learner a customized learning process. By sharing the other agents' experience in the CLE learning group, Agents can obtain additional materials. All these actions present a personalized learning support to the learner in online learning system.

An important issue to support effective cooperative learning is to describe all types of interactions between the agents. In the next section we discuss the Interaction levels of the agents in the learning environment.

4.7 Interaction

Four kinds of interaction between the related agents are thought out: interaction between the personal agent and the problem detect agent, between the personal agent and the motivation detect agent, between the problem detect agents each other in the learning environment, and between the motivation agents each other in the learning environment. The first two interactions are inside the personal learning environment, heterogeneous agents communicate each other to provide personalized support to the learner. In contrast, the last two are between the learners, homologous agents exchange their experience each other to enhance the learning progress.

Personal-Problem interaction: The personal agent interacts with the problem detect agent during the learning presentation. The problem detect agent could propose the recommended topics or information to the personal agent when it detect the learner has some difficulties on the learning topic. On the other hand, the personal agent provides the learner's personal information (learning history, navigation history, satisfaction of the presentation, etc) as the analysis material to the problem detect agent.

Personal-Motivation interaction: This is done with the personal agent and the motivation detect agent during the learning presentation. The motivation agent could prompt some additional presentation or dialogue to the personal agent when it detects the learner has some difficulties or strays from the learning topic. At the same time, the agent also requires the learner's personal information from the personal agent for its analysis.

Problem-Problem interaction: The problem detect agents share their experience to find out the learner's weakness and missing knowledge within a cooperative learning environment. In the cooperative learning environment, the problem detect agent can analyze the other agents learning history to help itself to detect the learner's problem or missing knowledge. The agent's suggestion dialogue could influence the progression of the lesson. It could recommend a new section or topic to the learner, or whether more exercises are needed upon the same topic, or even to decide a back-track to a previous topic.

Motivation-Motivation interaction: The motivation agents exchange their experience during the learning presentation. When the motivation agent tries to detect the learner's motivation, it not only tries to analysis the learner's history, but also consults the other learners' motivation action history. During the learning process, the agent could intervene when it detects a motivation difficulty of the learner and needs some motivation presentation. In additional, the agent could also suggest deepening some topics when it identifies the current topic too simple for the learner.

4.8 Overview

In this chapter we introduced the cooperative learning environment in the agent-base online system. We started out by explaining the essential feature of the cooperative learning environment. We also described the features and architecture of our PLE. After



the introduction of the mechanisms for cooperative learning, we ended this chapter with the interaction between the agents.

Prototype application

A web-base learning system was developed as a prototype application to apply the CLE into the online learning system. It's an online Java FAQ helping system which is implemented as a three-tier browser/server based application and can be smoothly deployed to a distributed learning environment over the Internet. In this chapter, we will represent the chief features of the application.

5.1 Three-tier application

Normally, the clients of the online learning application have various runtime environments and could be distributed all over the Internet. The front-end application's deployment and maintenance become a key distribution issue for this kind of application. The three-tier web-base application should be the easiest solution for now.

5.1.1 Definition

A three-tier application is a program architecture which is organized into three major disjunctive tiers. These tiers are

- Presentation tier (Front end)
- Logical tier (Middleware)
- Data tier (Backend)

Each layer can be deployed in geographically separated computers in a network. Some architects divide logical tier in to two sub tiers business and data access tiers, in order to increase scalability and transparency. The tiers can be deployed on physically separated

machines. The characteristic of the tier communication is that the tiers will communicate only to their adjacent neighbours. For an example, the presentation tier will interact directly with the business tier and not directly with data access or data tiers.

Following is a typical three-tier application architecture scenario.

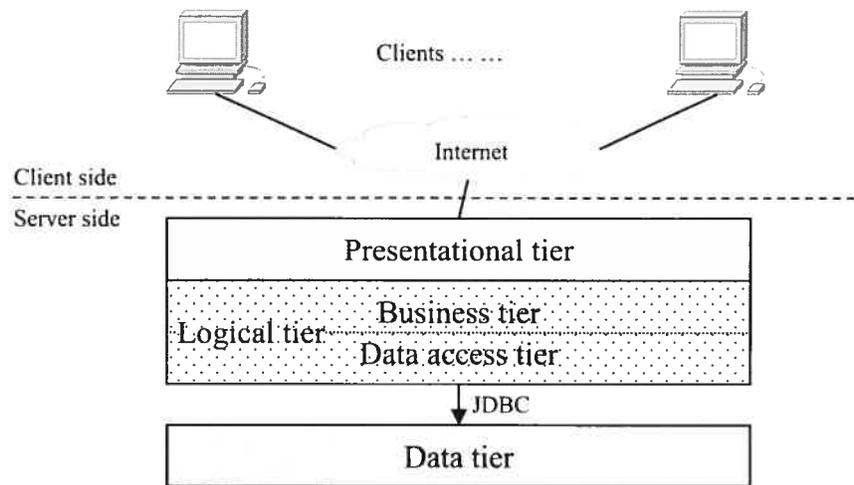


Figure 5-1: Typical three-tier application architecture

5.1.2 Two-tier versus three-tier

Client Server(C/S) architecture is a two-tier architecture because the client does not distinguish between presentation tier and logical tier. That is why we call this type of client as “fat client”. The increasing demands on GUI controls caused difficulty to manage the mixture of source code from GUI and business logic (spaghetti code). Further, C/S architecture does not support enough the change management. That makes it has lots of deployment and update issues. Let us suppose that the government increases the consume tax rate from 14% to 16 %, then in the C/S case, you have to send an update to each clients and they must update synchronously on a specific time otherwise you may store corrupt information.

The C/S Architecture is also a burden to network traffic and resources. Let us assume that about five hundred clients are working on a data server then we will have five hundred JDBC connections and thousand of record sets, which must be transported from the server to the clients (because the business logic tier is situated in the client side). The fact that C/S does not have any caching facilities like in J2EE server caused additional traffic in the network.

In the late 1990's, designers have shifted the business logic from the client to server to elude the handicaps from C/S Architecture. Normally, a server has a better hardware than client therefore it is able compute algorithms faster than a client, so this fact is also an additional pro argument for the three-tier architecture. On the other hand, we will never have the deployment and update issue because all the updates are implemented on the server side only.

Additionally, since the connections to the database server are only created when the client queries the data in the database, the server which handles five hundred clients will never have five hundred concurrent connections again. Information presents on the client side only the final result of the business logical.

5.1.3 Three-tier architecture

- **Data tier**

This tier is responsible for retrieving, storing and updating from Information therefore this tier can be ideally represented through a commercial database. We consider stored procedures as a part of the data tier. Usage of stored procedures increases the performance and code transparency of an application.

- **Logical tier**

This is the brain of the three-tier application. Two sub tiers include in it which are the business logical and the data access. The business tier contents classes to calculate aggregated values such like total visited items, most interested section. The data access tier will supply the needy information from the databases to the business logical tier. It acts as an interface to the data tier and knows how to (from which database) retrieve and store information.

- **Presentation Tier**

This tier is responsible for communication with the learners and it will use objects from business layer to response GUI raised events.

5.2 System architecture

The following figure shows the system architecture of the Java FAQ helping system (see Figure 5-2).

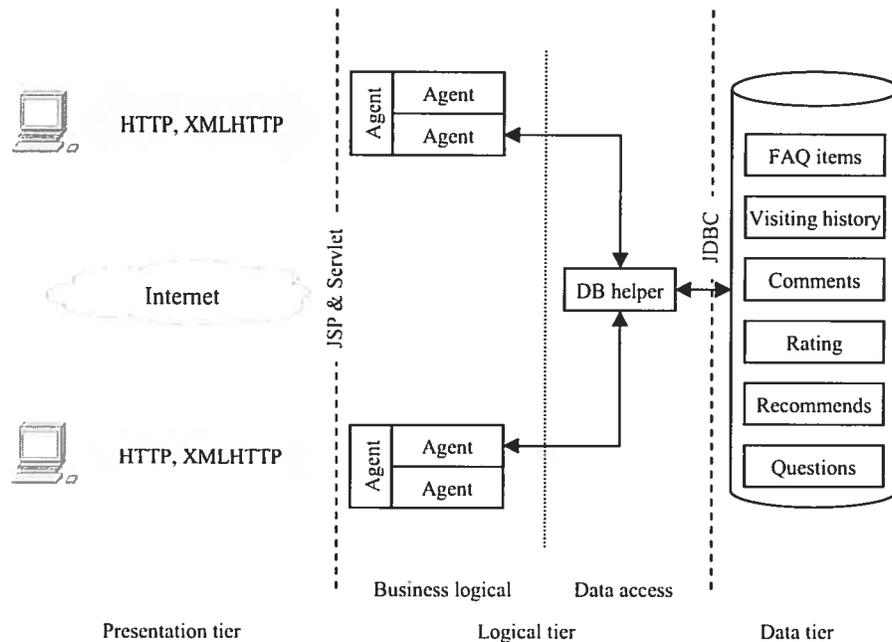


Figure 5-2: System architecture

Web browser is applied on the client side as the presentation tier. Communication between the presentation tier and the logical tier is through the standard HTTP protocol over the Internet. Because the http is a stateless protocol, a javascript-based client agent is included in the client side's page which provides the real-time observation for the learner.

JSP pages and servlets act as the interface modules to communicate with the learner. The business logical sub-tier contains the javabeans and the agents which implement the essential functionality of the system. Personal agent maintains the learner's profile and visiting history. Problem agent and motivation agent analyze the learner's behaviours during the learning session.

We also implement a group of data helper javabeans as the data access sub-tier to transparentize the database access process. Database interface class handles the query and

response between the system and the background. A set of data structure classes map the database record and table into independent data types.

At the background, we have a pure java SQL database server which service as the data tier provider. More detail description about the implementation is in chapter 7.

5.2.1 Presentation module

The presentation modules' structure is shown in the following figure. There are four key JSPs (FAQ, section, search and item pages) which generate the HTML to present the information to learner. Besides the FAQ content, they are all included some JSP fragments which present the learner's behaviours during the learning session (e.g. the latest visited topics, the previous questions).

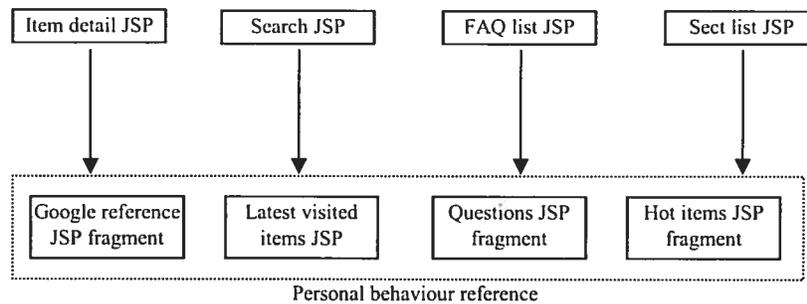


Figure 5-3: Presentation modules structure

5.2.2 Presentation modules reference

The following figure shows the reference between the JSP pages. FAQ list page is the default start page when learner creates a new learning session. From this page, learner can navigate to the other pages, except for the search page which has a servlet retrieving

the post back and redirects the result to the search JSP page. The item detail JSP page refers to three servlets to manage the learner's feedbacks about the FAQ topic.

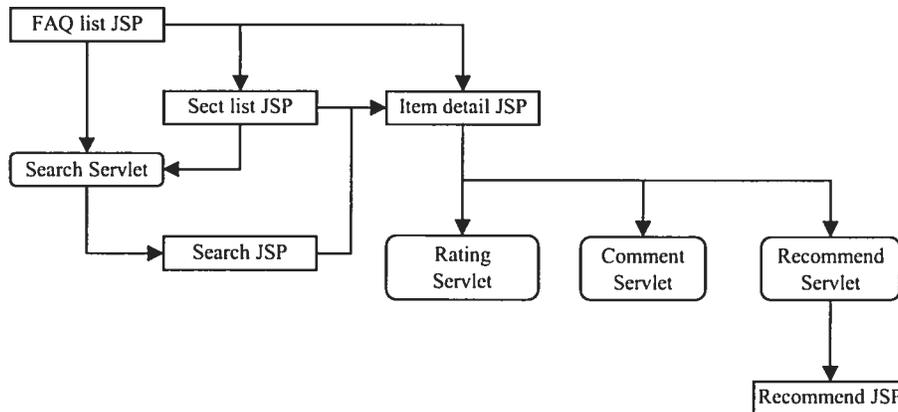


Figure 5-4: Presentation modules reference

5.2.3 Agents' structure

Agents' reference structure shows in the following figure. The client side present agent is included in every JSP page to collect the learner's real-time behaviour information and perform the agent's actions. Only the personal agent can communicate with the present agent, and the other two agents have to pass their message to the personal agent first to transfer it to the learner.

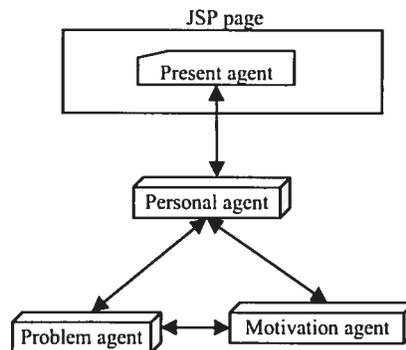


Figure 5-5: Agents structure

5.2.4 Data access structure

Data access sub-tier detaches the business logical and the actual database entity. A virtual database interface presents through a set of database independent data type class. Following figure shows the structure of the data interface classes.

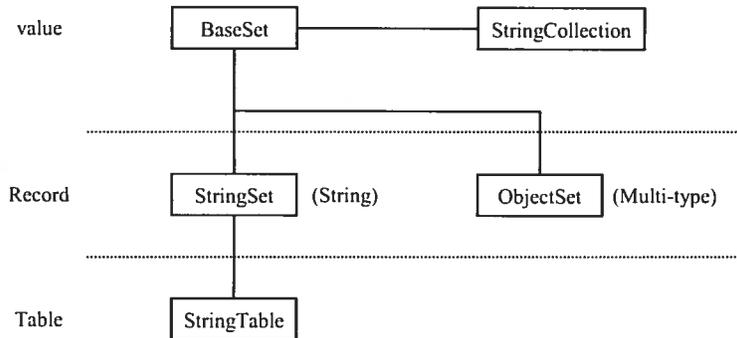


Figure 5-6: Virtual database interface

5.3 XML

Extensible Markup Language (XML) is a simple, very flexible text format derived from SGML (ISO 8879). Originally designed to meet the challenges of large-scale electronic publishing, XML is also playing an increasingly important role in the exchange of a wide variety of data on the Web and elsewhere.

To exchange the real-time client information, we use the XMLHTTP in the client agent to transfer the messages behind the JSP page. This protocol provides the ability to transfer the XML document through a HTTP channel in a browser/server environment. In addition, it avoids the web page being re-flashed when the client side agent communicates with the server side agents.

A sample XML document is shown as following:

Table 5-1: XML example

```
<?xml version="1.0" ?>
<Agents>
  <Tick>30</Tick>
  <ProblemAgent>
    <Suggest>You can post your question in the search box, or browse
the FAQ by categories.</Suggest>
  </ProblemAgent>
  <MotivationAgent />
</Agents>
```

The document demonstrates a member of the rules formed in a XML document. The first line in the above sample is the XML declaration, which defines the XML version of the document. In this case the document confirms to the 1.0 specification of XML. It's not a mandatory element of the XML document, but normally it should be included. All XML documents must have one enclosing element (except the version tag which does not count as an enclosing element). The root element of this XML document is the "Agents" which wraps the entire document and has to be the first element of the document. It contains three sub-elements (child nodes) – "Tick", "ProblemAgent" and "MotivationAgent". Because not one word in the XML document is an XML keyword, the most important for a freewheeling XML author is keeping the spelling in the tag names correct and make sure that each individual begin has an end tag.

The "Tick" tag informs the client side agent's next communication time. The "ProblemAgent" and "MotivationAgent" tag include the information from the problem agent and the motivation agent which could have three different behaviours – "Suggest", "Action" or "Animation".

Because the client agent aims to be a presentation agent at the client side, all the behaviours depend on the server side agents' XML document we present before. For instance, the "Suggest" behaviour could cause the client side agent popup the message

from the server side, and the “Animation” could cause the client agent perform some animation actions.

5.4 Data structure

The following figures show the data structure of the Java FAQ helping system.

5.4.1 FAQ item and it’s related tables

The FAQ items are stored in the “faq” table and they are organized into categories with the section number which links with the “sect” table. All the FAQ questions and answers were pre-analysed and the extracted keywords were stored in the table “keyword”. The “faqrate” table contains the learner rating information about the FAQ items and the queries. Learners’ comments of the FAQ are stored in the table “faqextra”. Learners’ visiting histories are stored in the “history” table.

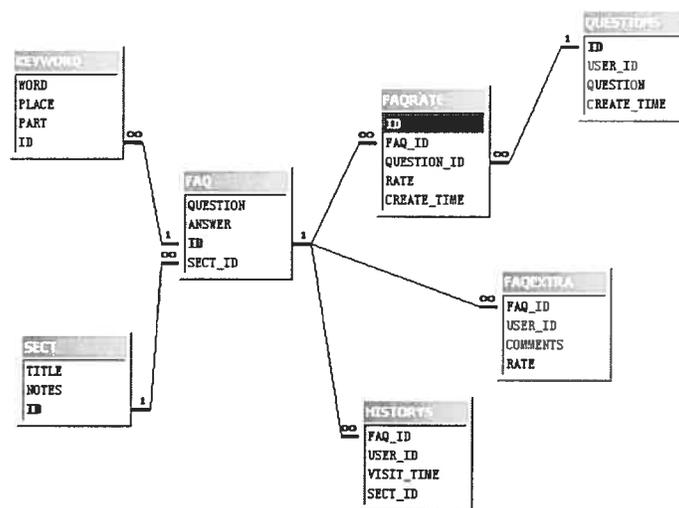


Figure 5-7: FAQ items and it’s related

5.4.2 Learner profile and its related tables

Learner’s profile table “user” is related with the learners’ behaviours tables, including the learner’s visiting history table “history”, learners’ question table “question”, learner’s comment table “faqextra”, and the learners’ recommendation table “recommends”.

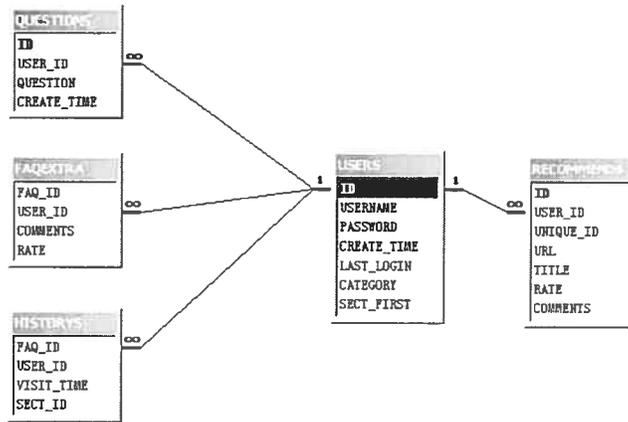


Figure 5-8: User table and it's related

5.4.3 FAQ category

All the FAQ items are grouped into 20 categories which are also the basic learner interesting group-base. Following table show the 20 categories.

Table 5-2: FAQ category

- Book Information
- General Information
- Compiler Messages
- I/O
- Computer Dating
- Swing
- Applets
- Networking and Distributed Objects
- C, C++
- Java GOTCHA'S
- Getting Started
- Compilers and Tools
- Java Language Issues
- Core library issues
- AWT
- Browsers
- Multi-Media
- Security
- Java Idioms
- Further Resources

5.5 Summary

In this chapter, we present the system architecture of our prototype application. We start by the introduction of three-tier application architecture which is used in implementation. After the presentation of the detail system modules structure, we explain the XML

documents structure which are used for the information exchange between the agents. We end this chapter with the description of the database structure.

Methodologies and algorithms

In online learning environments where there is the potential for losing the cohesiveness and spontaneity of the classroom experience, it is essential to understand how to improve the online learning experience so that it approaches and perhaps even exceeds more traditional instructional methods. The instant availability of a human tutor online would be ideal.

However, providing this capability is no more realistic than continuously providing a human tutor for the traditional classroom-based learning experience. Cost and availability are limiting factors in supplying continuously attentive human tutors in online learning. We think that agent can provide such tutorials and assistance for certain types of learning requirements. An augmented anytime capability is particularly important in online learning environments in which online tutors may not be available for extended periods (e.g., due to differences in time zones or to late-night student study habits).

6.1 Agent's responsibility

Online learning systems are networks of people who can learn anywhere and at anytime. The emphasis is on people learning with other people via the network. It has two core components – the people-to-people component as facilitated with computer conferencing, and a self-study part. The participation of agents bridges these components by providing help for the self-study part of online learning in a somewhat human way.

Researchers defined agents as computer programs that simulate a human relationship, by doing something that another person could otherwise do for you. For the purposes of online learning, our agent behaviours simulate what an expert could do, including the following characteristics:

- Provide rapid, accurate and useful advice whenever needed
- Be activated on-demand or whenever need is observed by the agent
- Encourage learners to complete learning session, to participate communication or other learning requirements.

6.2 Architecture

Figure 6-1 presents the logical architecture of our agent-based learning system. There are three basic components: the agents, the learner, and the knowledge base. As shown, the agents sit between the knowledge base and the learner, mediating the interaction. The internal architecture of the agents consists of personal agent, problem agent and motivation agent; all these agents compose a personal learning environment for the learner:

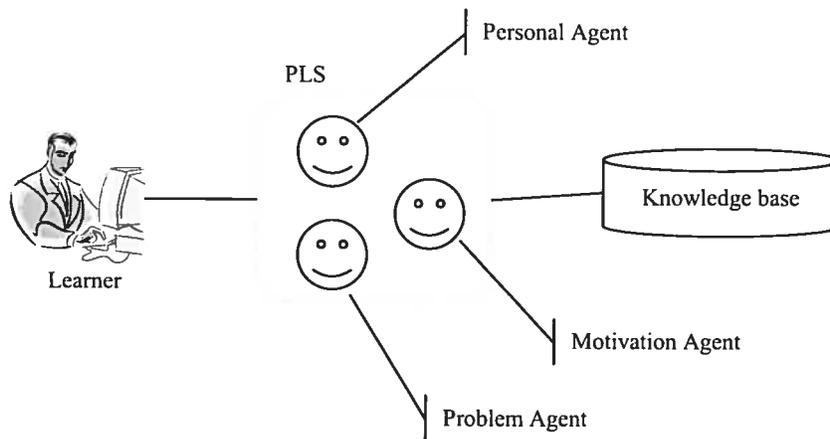


Figure 6-1: Logical Architecture

- Personal agents manage the learner's profile and act as a communication medium between the user and the other agents. Primarily, the functionality of the personal agents are to:
 1. Execute the other two agents by request
 2. Present information to the user
 3. Provide appropriate topic to execute actions such as requests for help
 4. Incorporate other relevant resources for the user
 5. Communicate with other agents (problem agent, motivation agent) and with the knowledge base (e.g., maintain the learner's learning history and behaviours).

- Problem agent is responsible for detecting learner's difficulty, searching and suggesting solution to the learner, and also forwarding the problem to motivation agent in some conditions. The difficulties include being trouble to handle a new learning topic, demonstrating a poor fund of knowledge, and rarely interacting with other learners. Problem agent detects these difficulties by analyzing the learning history and comparing with the similar learner within a same topic.

- Motivation agent is responsible for learner's motivation difficulties, dialoguing with the learner to prompt appropriate resource. The agent can be invoked either by personal agent or by problem agent. The main functionality of the motivation agent is to determine the completion status of the topic base on the learner's behaviours. Moreover, by comparing the individual learner's behaviours, the motivation attempts to determine what additional resource could the participant in order to complete the topic. The agent provides extended knowledge based on the result of analyzing the other learners' commentary and recommendations.

6.3 Problem situation

Detecting the problem situation is the very beginning of all the agents to help the learner. In the PLS, we categorize the problem situations as having affective, cognitive, structural and interpersonal difficulties by the pedagogical state during the learning [Huang 2003].

- Affective difficulty

Learner is in an affective difficulty situation when he has trouble to handle important events, such as new phases of their education. This difficulty in adjusting may lead to affective reactions that ultimately manifest as difficulties with motivation. Typically, for the new learner of the system, this difficulty happens frequently.

- Cognitive difficulty

Learner is in a cognitive difficulty situation when he has trouble in communication or material integration. They may fall behind in workload, demonstrate a poor fund of knowledge, or perform poorly in discussions. Learners with continued cognitive difficulties may have an underlying learning disability.

- Structural difficulty

Learner is in a structural difficulty situation when he has trouble to structure their experiences in the learning environment. They may demonstrate poor time management and disorganization by spending a long learning time on the topic.

- Interpersonal difficulty

Learner is in an interpersonal difficulty situation when he does not interact well with other people, for instance, the other learners in the learning environment. They may have either a mild disorder characterized by shyness or poor social skills or a more severe disorder in which they are manipulative or confrontational.

6.4 Problem detecting routine

For a new learner of the learning system, probably he is not familiar with the system, or he is a raw recruit of Java programming, whichever of these problems will bring him into some kinds of affective difficulty situation. Learner could be confused by some of the pages for an extremely long time before he finds out the information he needs. Or the “Can’t understand” is feed back for most the search results. The problem agent would detect that the learner is in the effective difficulty when the above behaviours happened.

As the description in the preceding session, effective difficulty usually manifests as difficulties with motivation. When the problem agent catches this difficulty problem, it passes it to the motivation agent to interview the learner. The question that motivation agent posted to Learner could be similar to: “You have read the FAQ list page for a long time. Can I help you?” and than the agent will give some options to identify to the learner’s motivation. Like: 1) Please give me an introduction of how to use the FAQ helping system. 2) Where’s the best place to start using the FAQ system?

6.4.1 Problem agent action’s process

Problem detecting process presents a standard procedure to deal with the behaviour analysis and problem routing. It includes three steps:

1. Type and specify the ineffective behaviours, redirect these behaviours, provide a more detail description about the behaviours to improve the learner.
2. Identify the category of difficulty by the learner, using the description of the different types of problem situations mentioned above. This step is important since planning a strategy to help to learner depends on an accurate assessment of the difficulty situation. Once the situation has properly categorized;
3. Asking the learner’s opinion about the answer.

Step 1 and step 2 are executed by the problem detect agent. If the problem agent can find out a solution base on the suggestion, it will directly post back to the personal agent to display. Otherwise, the problem will be transferred to the motivation agent to start a conversation with the learner which is the step 3.

6.4.2 Affective difficulty reaction roles

Following, we provide a detail description about the affect difficulty reaction as an example of the detecting process.

Behaviour: learner puzzles on a page

Location: “FAQ list” page

Question from the agent: Are you still there? You have read the page for a long time. Can I help you?

Options that motivation agent prompts to learner:

- 1) Please give me an introduction of how to use the FAQ helping system
- 2) Please suggest a category

Actions: When option 1 is selected, motivation agent will transfer the learner to the introduction page. Otherwise, according to the learner’s behaviour history: If he never visits his interesting category (chosen when the learner first time logins), suggest the learner browse that category, else suggest the learner visits the most related category, and the learner visits the very little visited category.

Location: Search result list page

Asking: Can’t find the questions and reference you need?

Actions: redirect Learner to the search suggestion page.

Behaviour: learner feeds back a low rating for the FAQ item (only on the FAQ item information page)

Question from agent: the reason of the low rating

Options:

- 1) Too easy
- 2) Useless
- 3) Not clear enough
- 4) Can't understand

Actions:

- 1) Learner has enough knowledge to understand the answer and needs in-depth explanation. Try to find some advance question in the same topic. Motivation agent requests the Google searching engine to provide additional resource about the topic.
- 2) Learner understands the answer but it's not the facet he concerned. Try to find another related question and answer. Problem agent suggest learner to extend or change the search query to relocate the learning topic and category.
- 3) Learner finds the answer helpful but it's not clear enough for him. Need more detail about the answer. Suggest the reference. Problem agent suggest learner to check the reference documents from Google engine and/or the other learners' recommendation.
- 4) The answer is too difficult to the learner. Need basic information/knowledge about the question and answer. Suggest the definition. Motivation agent prompt learner to use the definition search tools inside the interface.

Behaviour: learner rarely participate the topic discussion

Action: motivation agent prompts learner that all the discussions are anonymous, and describes the advantage of the online discussion.

6.4.3 Interpersonal difficulty reaction roles

The CLE application provides certain ways to interact the learners each other. For instance, learner can comment a question, recommend a reference page to the other learner, and rate a question by his comprehension. All these behaviours are offered to help the learner interact each other in the CLE.

If he doesn't use them totally, or the frequency of using these functions is extra lower than the average, the problem detect agent would suppose the learner is in the interpersonal difficulty. The agent will present the functionality and the benefits of these interactions to the learner, advise the learner to use them in the CLE.

After that, if the difficulty still exists, the motivation agent will converse with the learner. By asking the learner the questions, the agent tries to find out the reason why he doesn't use them and suggest some solutions. The probably reasons include:

1. The learner needs more detail introduction about the communication tools. Advise him to read the tutorial of the system.
2. The learner could be shyness. Explain the learning environment is anonymous and opening.
3. The learner doesn't have enough experience in the CLE.

6.5 Learning from Feedback

Learner can communicate with the agents by providing feedback about interesting topic in two ways. The first way is to provide positive or negative feedback for the topic retrieved by the agent. Secondly, the learner can provide feedback for the Google reference documents. In each of these cases, learner feedback has two effects. One is that the appropriate profile is modified in response to the feedback, especially the learner's

affective and interpersonal behaviours. The other is that the fitness of the topic responsible for the search is appropriately modified.

Relevance feedback has been used to improve the performance of retrieval systems [Salton and Guckley 1990]. For a weight estimation system, the method for query reformulation in response to learner feedback is weight adjustment. The weight of the topic related with the query is increased when the agent received positive feedback and decreased when the agent received negative feedback. And for the Google reference web page, we use the same mechanism to evaluate the weight for the query.

A query - topic related weight estimation system is used to optimal the search result in the future. In this system, the weight of the topic is created and related with the query which means the topic's weight could be different for different queries. For a query, the search result is not only decided by the general factors, e.g. the keyword's frequency, but also the topic weight which also influences its display order in the query result. This feature makes the query result could be dynamic even for the fixed FAQ database topics.

The feedback mechanism used in our learning system is a generalization of the above method. The topic weight for the specific query is modified in response to user feedback. The process of modification for each of the topic weight is similar to the classical weight estimation method.

6.6 Vector-based similarity weight measure

In this section, we introduce our vector based similarity weight measure which develops from the basic static vector weight measure of information retrieve. Basically, the distance d between the documents D_i and the query q is defined as the following formula.

$$d(D_i, q) = \sum_{i=1}^n freq(D_i, w_k)$$

Equation 6-1: Basic distance

where:

w_k : k^{th} keyword in the query q .

$freq(D_i, w_k)$: The frequency of word w_k in D_i .

In general, this formula is precise enough for the query which includes just one keyword. However, for the queries which have more than one keyword, the order and clustering level of the keywords has to be considered during the document clustering. Following, we represent a set of the arguments to evaluate the clustering level of the query's keywords in the documents.

Step distance – the minimum distance between two continual query keywords in the document. For the completely matched word sequence, the step distance of word is zero.

For word w_i in the document D , the step distance sd_i is:

$$sd_i(D, q) = \min(loc(w_i) - loc(w_{i-1}))$$

Equation 6-2: Step distance

where:

$loc(w_i)$: The location of word w_i in document D

Distance - the total steps distance between the matched keywords in the document. Specifically, for the document which has a completely matched word sequence, the distance should be zero. For query q which has n keywords, the distance of document D is:

$$d_2(D, q) = \sum_{j=2}^n sd_j(D, q)$$

Equation 6-3: Document distance

Closest distance - We define the closest distance as the minimum step distance for all the matched words in the document. For query q which has n keywords, the closest distance of document D is:

$$d_3(D, q) = \min_{j=2}^n sd_j(D, q)$$

Equation 6-4: Closest distance

Continual length – For the matched word sequence which step distance is zero, we calculate the continual words length in the sequence to show the words' clustering.

$$cl(D, q) = \max len(D, q)$$

Equation 6-5: Continual length

Where:

$len(D, q)$: the continual matched words length in Document D

According to the arguments above, the total weight W of document D for query q is the functions all these arguments with their adjusting weights.

$$W(D, q) = \alpha \cdot d(D, q) + \beta \cdot d_2(D, q) + \eta \cdot d_3(D, q) + \mu \cdot cl(D, q)$$

Equation 6-6: Document weight for query

Where:

α, β, η, μ : the adjusting weights for all these arguments

6.7 Summary

In this chapter we introduced the learning difficulty situation in the learning process. We have divided this kind of situation into four types which are affective, cognitive, structural and interpersonal difficulties. We also discussed the methods that we used in our cooperative learning environment to detect and avoid the difficulty situation. Additionally, we presented the weight estimation algorithm and the search strategy we used in our CLE. We ended this chapter with the problems related with the motivation detection which the problem has to communicate with the motivation agent.

Implementation

This chapter discusses the Java JAQ helping system, a prototype application implemented based on the algorithms presented in the preceding chapters.

In order to illustrate the architecture and methodology of our CLE, we constructed a Java FAQ helping system – an online multi-agent cooperative FAQ learning system. Our database includes 400 FAQ items (the questions and answers) which organized into 20 categories (details in the Chapter 5).

7.1 Run-time environment and development tools

In this section, we introduce the development environment and database used in our application.

7.1.1 Development environment

A J2EE container application server is used on the server side which is a platform for designing, developing, debugging, distributing, implementing and managing the Internet-based helping system. Compared to traditional techniques, an application server provides extensibility and stability for the application. Certainly, the application server technique is not an essential element in our CLE, any other application of Internet-based accessing data mainly by web browser also can be used as the application server including most of the open source products and commercial software.

Tomcat is the servlet container that is used in the official reference implementation for the Java Servlet and Java Server Pages technologies. The Java Servlet and Java Server Pages specifications are developed by Sun under the Java Community Process. The version 5.x releases implement the Servlet 2.4 and JSP 2.0 specifications.

Servlets are the Java platform technology of choice for extending and enhancing Web servers. Servlets provide a component-based, platform-independent method for building Web-based applications, without the performance limitations of CGI programs. And unlike proprietary server extension mechanisms (such as the Server API or modules), servlets are server- and platform-independent. This leaves you free to select an appropriate strategy for your servers, platforms, and tools.

Servlets have access to the entire family of Java APIs, including the JDBC API to access enterprise databases. Servlets can also access a library of HTTP-specific calls and receive all the benefits of the mature Java language, including portability, performance, reusability, and crash protection.

JSP technology is an extension of the servlet technology created to support authoring of HTML and XML pages. It makes it easier to combine fixed or static template data with dynamic content. Java Server Pages (JSP) technology enables Web developers and designers to rapidly develop and easily maintain, information-rich, dynamic Web pages that leverage existing business systems. As part of the Java technology family, JSP technology enables rapid development of Web-based applications that are platform independent. JSP technology separates the user interface from content generation, enabling designers to change the overall page layout without altering the underlying dynamic content.

JSP technology uses XML-like tags that encapsulate the logic that generates the content for the page. The application logic can reside in server-based resources (such as JavaBeans component architecture) that the page accesses with these tags. Any and all formatting (HTML or XML) tags are passed directly back to the response page. By separating the page logic from its design and display and supporting a reusable component-based design, JSP technology makes it faster and easier than ever to build web-based applications.

Java server pages technology and servlets provide an attractive alternative to other types of dynamic Web scripting/programming by offering: platform independence; enhanced performance; separation of logic from display; ease of administration; extensibility into the enterprise; and, most importantly, ease of use.

7.1.2 Database engine

A pure Java relational database is used to store our FAQ items and other data. The HSQLDB is a platform independent relational database engine 100% written in Java, with a JDBC driver, supporting a rich subset of ANSI-92 SQL (BNF tree format). It offers a small (less than 300k), fast database engine which offers both in memory and disk based tables. The engine supports two different run-time modes which are embedded and server modes.

The embedded mode runs the database engine as part of your application program in the same Java Virtual Machine. For some applications this mode can be faster, as the data is not converted and sent over the network. In this mode, the application cannot connect to the database from outside your application. The embedded mode database is started from JDBC with the database file path specified in the connection URL.

On the contrary, the server mode provides the maximum accessibility. The database engine runs in a JVM and listens for connections from programs on the same computer or other computers on the TCP/IP network. Several different programs can connect to the server and retrieve or update information. Applications programs (clients) connect to the server using the HSQLDB JDBC driver. Additionally, it includes tools such as a minimal web server, in-memory query and management tools (can be run as applets) and a number of demonstration examples.

The recommended way of using this mode in an application is to use an HSQLDB Server instance for the database while developing the application and then switch to In-Process mode for deployment.

7.2 Class diagram

In this section we present the important classes used in our application, we will focus the workflow related classes. For other classes such as servlet related classes, we will discuss with the user interface section.

7.2.1 User class

The user class maintains user profile information. It also provides references to some of important classes in the CLE application, e.g. the database helper class, the Google search class and the FAQ item class. Figure 7-1 display the user class structure and its relationship with other classes in the application.

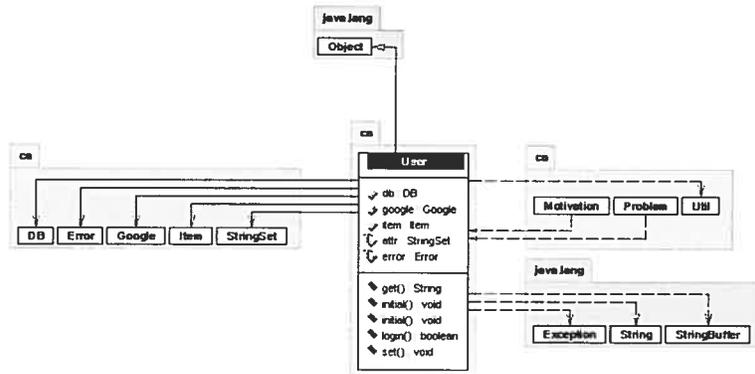


Figure 7-1: User class and its related

7.2.2 Item class

This class is an in-memory entity of the FAQ item which is created dynamically during the session. Major part of the information comes from the “FAQ” table; the other is user related and action related. For instance, the previous and next items, they are the previous and next item in the same category in browse mode rather than the items in the search result set in the search mode.

The item class contains the information including a FAQ item and the related access information, e.g. the item’s section, the previous and the next item. It also provides the learner’s participation information for this item, e.g. the comments for the item, the rating information. In addition, it offers function to update the learner’s visiting history. Figure 7-2 displays the item class structure and its related classes.

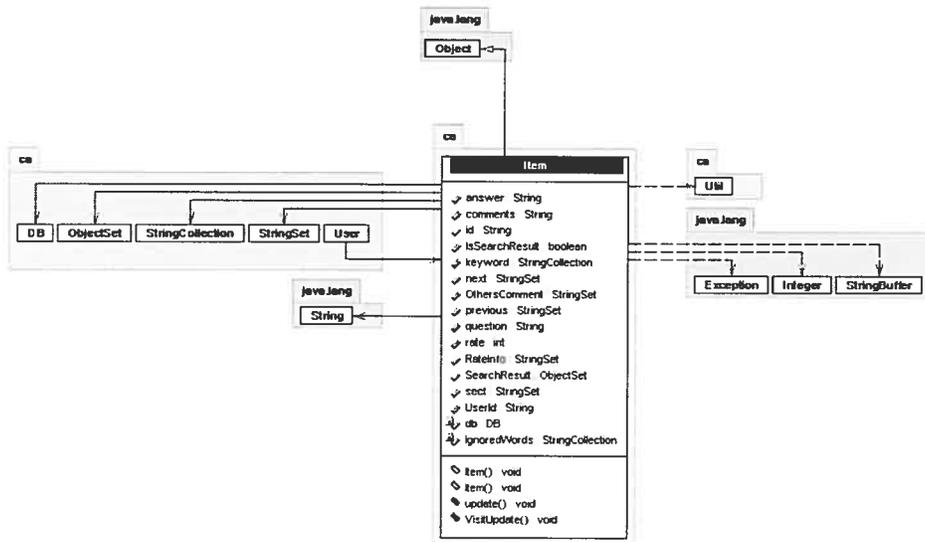


Figure 7-2: A FAQ item and its related

7.2.3 Database helper class

Database helper class provides a transparent database access sub-tier to the application. The other classes can access the database through this simplified database interface without considering the deference between the database products. In our application, all the database operations are through this class which provides the ability that we change the specific database engine and does not influence the business logical modes. Figure 7-3 display the data access class structure and its related classes.

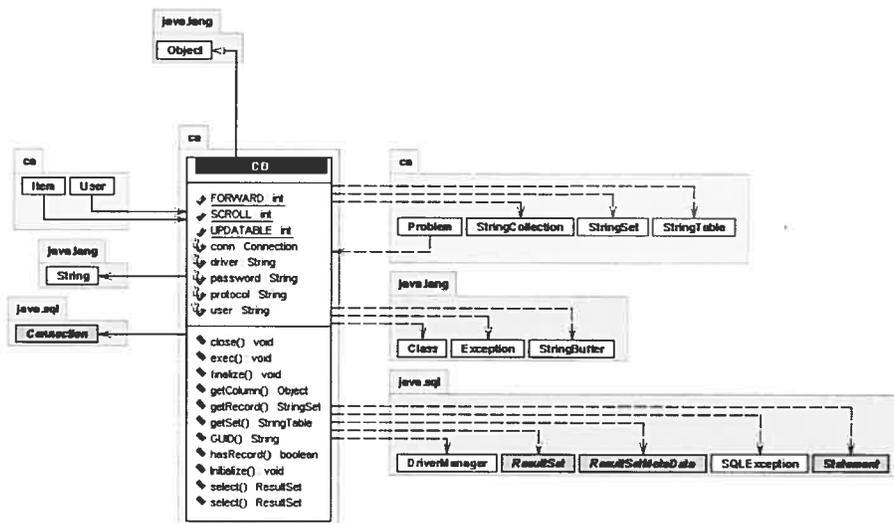


Figure 7-3: Data access helper class

7.3 Scenario diagram

To fully understand the underlying architecture of our prototype application – a Java FAQ helping system, both static and dynamic analyses are needed. In chapter 5, we have introduced the architecture of the system which is based on the static analyses; we also presented the class diagrams in the preceding section. Since the learners' browsing is a stateless behaviour, dynamic analyses are especially important for understanding the runtime behaviour of the learners in a distributed online learning system. In the following sessions, we select four important scenarios from the application to present the learners' browsing behaviour in the learning system. They are browsing the FAQ list, searching the FAQ, reading the FAQ item and recommending a reference.

7.3.1 Browsing scenario

The following figure shows the learners' browsing behaviours (see Figure 7-4). After the learner logs in the FAQ helping system, he will go to the FAQ list page (step 2) which is the first page of the helping system. From there, the learner can select a section to browse its questions (step 3). The section list page presents all the questions under its topic and the Google reference. The learner can select one of the questions to open the FAQ item display page (step 4), or picks up a reference document from the Google search result and then open the reference page (step 5). The Google reference information also exists in the FAQ list page which is about the FAQ entirely. Learner can also pass the section list page (step 3) and item page (step 4), and open the reference page (step 5) from FAQ list page (step 2) directly.

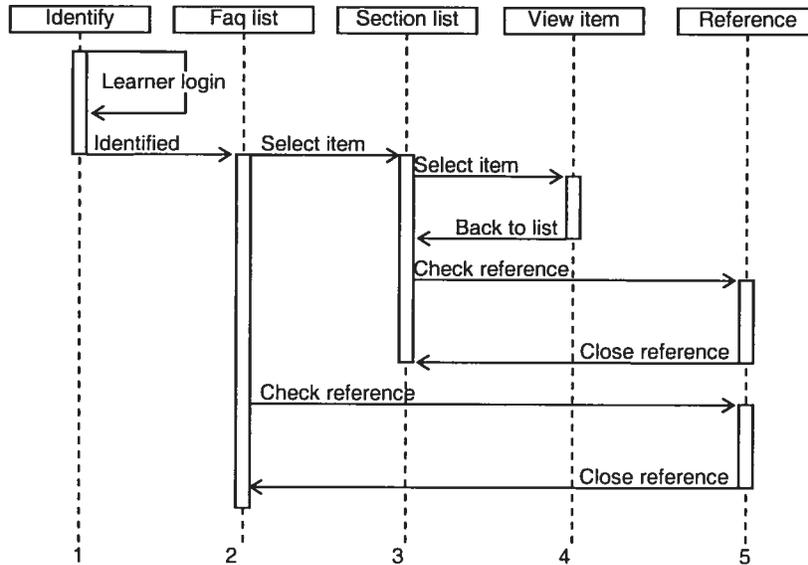


Figure 7-4: Browse scenario

7.3.2 Search scenario

The following figure shows the learners' searching behaviours (see Figure 7-5). A search can be launched from the FAQ list page (step 1), the section list page (not in the figure), or the search result page itself (step 3). Learner types the question in the search box and submits it, or selects a former asking question from the question list. When the search servlet (step 2) receives the search form's post-back, it performs the searching in the FAQ database. Instead of presenting the result to the learner from the servlet itself directly, the search servlet redirects the result to a search result page (step 3).

The agents in the environment are notified by the result page (step 3) to analyze the learner's learning state and a Google reference information list also presents in that page. The learner can pick the matched search result item in the result page (step 4), or check the reference from Google (step 5).

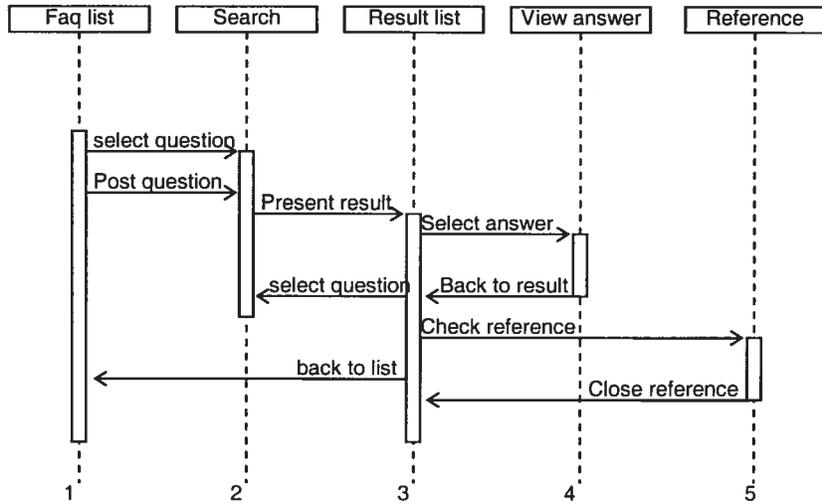


Figure 7-5: Search scenario

7.3.3 Reading scenario

The following figure shows the learner's reading behaviour. When learner opens the FAQ item viewing page (step 1), he can rate the question's answer (step 2), or post a comment about the question (step 3), or open a reference page of the question (step 4).

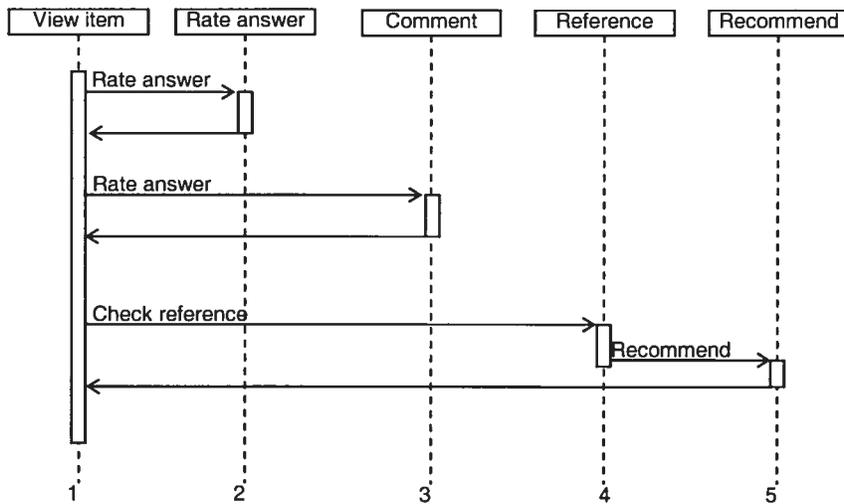


Figure 7-6: Reading scenario

7.3.4 Recommending scenario

The following figure shows the learner's recommending behaviour. Reference list (step 1) is included in most pages during the learning session (e.g. FAQ list, section, search, etc). When a learner finishes reading a reference (step 2), a popup dialog ask him/her to recommend the reference. Learner can select close the reference or open the recommendation dialog (step 3). In the recommendation dialog, learner can rate the reference satisfy level and make a comment for it. A servlet (step 4) will handle the post-back form to update the recommend table.

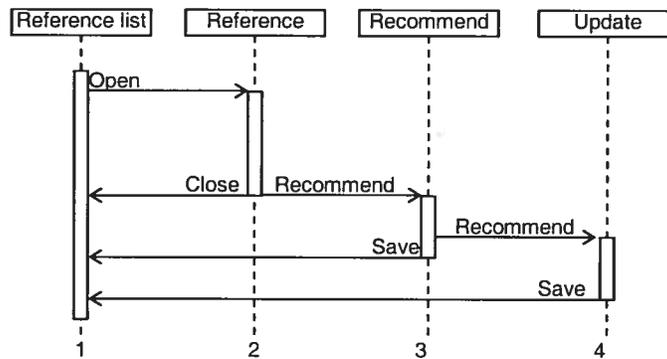


Figure 7-7: Recommending scenario

7.4 User interface

In order to better present the Java FAQ helping system, the following sections will introduce the user interface of the learning system.

7.4.1 Login & Register

Before learner can access the web site, he/she must register himself as a user of the system. The application applies an automatically user generation strategy. The user's profile will also be created automatically when learner logs first time, and there's not separated registration page in the application. Figure 7-8 displays the login page interface.

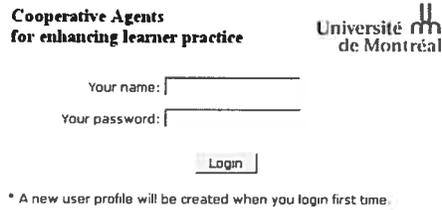


Figure 7-8: Login

7.4.2 Select the interesting section

When a new learner profile is created, the system asks him the interesting category which will be used as the short term motivation factor. Figure 7-9 displays the interface.

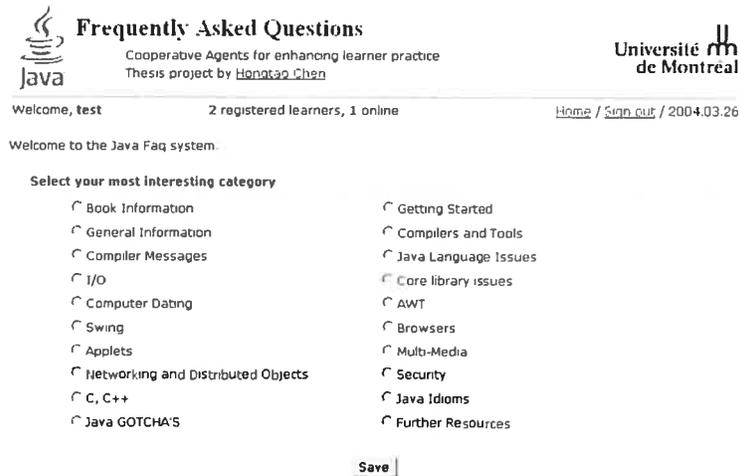


Figure 7-9: Select the interesting section

7.4.3 FAQ list

FAQ list page is the default start page for learners. In this page, learner can browse the sections of the FAQ database, post query to the search agent to find out the matched FAQ items. Some important visiting history also lists in the page, including the asked questions of the learner, the latest visited FAQ items of the learner, this most visited FAQ items for all the learners. A Google reference about the Java FAQ also includes at the bottom of the page.



Agent tip: Category "[Getting Started](#)" is a good place to start learning the FAQ.

Post your question |

Browse the Java FAQ by category

- [Book Information](#) (1)
- [General Information](#) (22)
- [Compiler Messages](#) (14)
- [FAQ](#) (24)
- [Computer Dating](#) (37)
- [Swing](#) (18)
- [Applets](#) (17)
- [Networking and Distributed Objects](#) (21)
- [C++](#) (10)
- [Java GOTCHA'S](#) (25)
- [Getting Started](#) (25)
- [Compilers and Tools](#) (18)
- [Java Language Issues](#) (31)
- [Core Library Issues](#) (20)
- [AWT](#) (41)
- [Browsers](#) (8)
- [Multimedia](#) (21)
- [Security](#) (9)
- [Java Idioms](#) (20)
- [Further Resources](#) (24)

Figure 7-10: FAQ list

From this page, a client side javascript agent is loaded at the page's background to present the server side agents' real-time information to the learner. The client agent will stay at the tray-bar until it gets a new message to display it to the learner, or it receives an order from the server side to execute an action. The client side agent also collects learners' environment information to the agents. Figure 7-11 presents a screen shot when the client agent receives a message from the server, it pops up and displays the message.

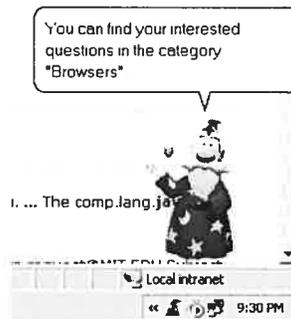


Figure 7-11: Client side agent pops up to display a message

7.4.4 Section list

Section list page presents the question list of a specific category. This page also includes the information of the learner's visiting history, e.g. the asked question and the latest visited FAQ items. Learner can navigate to the previous or the next category by the links

showed after the question list. The Google reference for the sections also includes in this page.

Frequently Asked Questions
Cooperative Agents for enhancing learner practice
Thesis project by [Hongtao Chan](#)

Université de Montréal

Welcome, test 3 registered learners, 1 online [Home](#) / [Sign out](#) / 2004.03.26

Applets

1. What is the difference between an application, an applet, and a servlet?
2. My applet works on my machine, but fails when I put it on our web server. Why?
3. How do I load a webpage using an applet?
4. How do I use an image as the background to my applet? How do I set the background color of my applet the same as the browser?
5. How do you make the applet's background transparent?
6. How do you do file I/O from an applet?
7. How do I pull a non-class file, such as a .gif, out of a jar file?
8. How do I read a text file stored in a Jar?
9. How do you get a Menubar/Menu in an applet?
10. Can I get rid of the message "Warning, Applet Window" along the bottom of my popup windows in my Applet?
11. When I subclass Applet, why should I put setup code in the init() method? Why not just a constructor for my class?
12. I want to know about {applets,applications} but the lousy book I got just talks about {applications,applets}. What can I do?
13. How do I print a page with an applet?
14. How can I position my dialogs centered (not top left)?
15. How can I get two applets on the same page to communicate with each other?
16. How can I resize an applet?
17. How do I sign an applet?

<< [Browsers](#) [Multi Media](#) >>

Figure 7-12: Section list

7.4.5 Question & answer

Question & answer page present the detail information of a FAQ item. Learner can rate the quality of the answer in this page. A comment text area also includes in this page in which learner can describe his opinion and comprehension about the question and answer. If other learners have comments about this item, it will also displays after the comment text area. Like the other pages, this page contains the learner's visiting history information, and the Google reference about this item shows at bottom of the page.

If learner navigates from a search page, the hyper link will appear in front of the FAQ question from where learner can return the search result list to pick up another matched item. In the page, the FAQ's question and answer are highlighted with the matched keyword in them. Additionally, the item's navigation hyper links at the bottom of page is also modified. Normally, they are the previous and next items in the current item's

category, but if it's a search result, they are changed to the previous and next items in the search result list.

Results for the query: [remote call](#) Category: [Not-working and Distributed Objects](#)

Should I use CORBA in preference to RMI? Or DCOM? Or what?

If your distributed programs are wholly written in the Java programming language, then RMI provides a simpler mechanism that allows the transfer of code, pass-by-value of objects, and automatic garbage collection of remote objects.

If you need to connect to C++ (or other language) systems or you need CORBA-specific services, then CORBA is your choice.

In Java 1.3 Sun has aligned RMI to work more closely with CORBA. Sun has simply added an IIOP transport layer to RMI to support interoperability with CORBA. Java technology-enabled programs can now use RMI to access CORBA-based objects through IIOP, the OMG's CORBA-based protocol. This is very good news for those building heterogeneous Enterprise systems, although it will take some additions to IIOP to support the pieces that RMI uses.

Microsoft spokespeople have tried to promote DCOM by spreading misinformation that RMI is changing or being dropped. That is totally wrong. The RMI API continues unchanged in its current form. Using DCOM would restrict your code to only ever run on Microsoft platforms using Intel hardware, and negates the "write once, run anywhere" Java philosophy. You would have to recompile your DCOM code to run it on other Microsoft platforms like Compaq's (formerly DEC's) alpha computer. Non-portable, single vendor code should be avoided. DCOM/DNA has limitations for use in the enterprise.

Other sites:

<http://www.java-brid.com/java-bridge-10-1997/tw-10-corbajava.html> has a good intro to CORBA in the Java world.

<http://www.objects.com/katus> has a CORBA/RMI comparison.

Rate the answer

Outstanding Poor

Comment this question

Figure 7-13: FAQ item - question & answer

7.4.6 Search result

Search result page displays the matched items of learner's query. All the matched keywords in the questions are highlighted which make the learner could easily find them in the list.

Instead of using the search result page directly, the learner's query is posted back to a search servlet which performs a dynamic weight estimation search (detail in Chapter 6) to find out the matched items in the FAQ database. After the search, the search servlet saves the search result and redirects the output stream to this search result page to present it. When this page presents the search result, it also queries the Google web service to obtain the Google's reference about the learner's question.



Results for the query: **convert date**

1. [How do I convert a date in one format to another?](#)
2. [What happened to java.util.Date between JDK 1.0 and JDK 1.1?](#)
3. [How can I convert between GIF and JPEG formats?](#)
4. [How do I calculate the number of days between two dates?](#)
5. [When I print a java.sql.Timestamp it doesn't include any milliseconds. What is the problem?](#)
6. [What timezone does a java.sql.Date use when converting to an SQL_DATE?](#)
7. [How do I use a DateFormat to create a text string from a Date?](#)
8. [How do I use DateFormat to parse a string containing a date?](#)
9. [The GregorianCalendar will not accept a date prior to 4713 B.C. Why?](#)
10. [Should I stop using Date all together and just use Calendar?](#)
11. [When I create the date July 4th, 1776 using new GregorianCalendar\(1776, 7, 4 \) the month is off by one. What is the problem?](#)
12. [How do I use a GregorianCalendar to extract a few fields from a Date?](#)
13. [How do I create a specific date in the Gregorian Calendar?](#)
14. [Since Date\(int year, int month, int date\) and related constructors are deprecated what do I use instead?](#)
15. [Since the Date\(String \) constructor is deprecated what do I use instead?](#)
16. [I really don't need an internationalizable, timezone aware, extra flexible formatting set of date classes. Is there anything else I can use that stores only a date, and allows me to do some date calculations?](#)
17. [Why are all the methods in java.util.Date deprecated?](#)
18. [How do I create a Date object that represents the current time?](#)
19. [Exactly what is a java.util.Date?](#)
20. [What is "Copy Bird" and where can I get it?](#)

Google reference information

- [Java Technology Forums](#)

Figure 7-14: Search result

7.4.7 Recommend a reference

Every time, after learner read the Google's reference page, a popup dialog will ask him to recommend the document. Figure 7-15 is the interface of the dialogue.



Figure 7-15: Asking recommendation

The reference document web dialog (Figure 7-16) displays the reference page's title and URL information. Learner can rate the quality of the reference and made a comment about it.

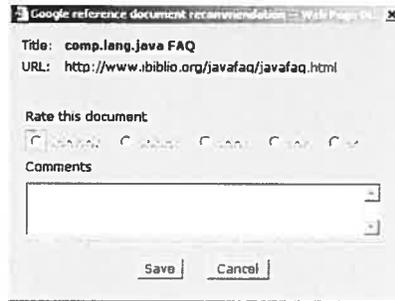


Figure 7-16: Recommendation web dialog

7.4.8 Customized mouse right click menu

Instead of using the standard mouse right click menu in the web browser, our system implements a customized right click menu (Figure 7-17) on the pages. This right click menu provides some convenient functions to assist learners using the FAQ system. In this menu, learner can return to the start page, get the latest search result, or sign out the system.



Figure 7-17: Customized mouse right click menu

One of the most practical features that the menu provided is finding the word's definition in the page. Learner can select any text in the page and then use the "Find definition" to search the words definition in an online dictionary (www.hyperdictionary.com). When learner doesn't have enough background knowledge to understand the FAQ question or answer, the feature can help learner to work out the difficulty in text. The following figure (Figure 7-18) is an example of definition searching.



English Dictionary Computer Dictionary Thesaurus Dream Dictionary Medical Dictionary

Search Dictionary: Remote Method Invocati Search

Learn new languages

REMOTE METHOD INVOCATION: Dictionary Entry and Meaning

Computing Dictionary

Definition: [PM] Part of the Java programming language library which enables a Java program running on one computer to access the objects and method of another Java program running on a different computer.

Figure 7-18: Find definition

Another useful feature in the right click menu is searching the page's content text in Google engine. Learner can select any text in the CLE interface and then use the "Search in Google" to find out the Google's reference web pages for the selected text. The following figure (Figure 7-19) is an example of using Google search engine.

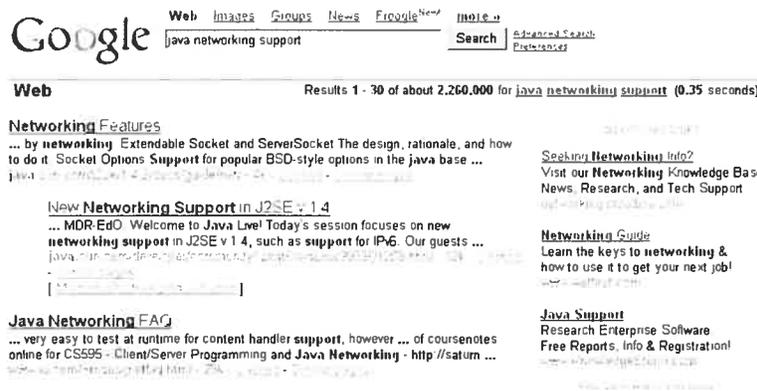


Figure 7-19: Search in Google

7.5 Examples & Scenarios

In this session, we present some examples in the application to demonstrate the main features of the system, including the personalized support feature and the cooperative learning advantage.

Scenario 1: Start Page

When Learner logs the application, in the start page, learner can navigate inside the category list or type his/her question in the search box to find out the result. In the meantime, the personal agent will prompt the learner to use the search box to quickly find out his answer when the learner stays on that page for several minutes (Figure 7-20).

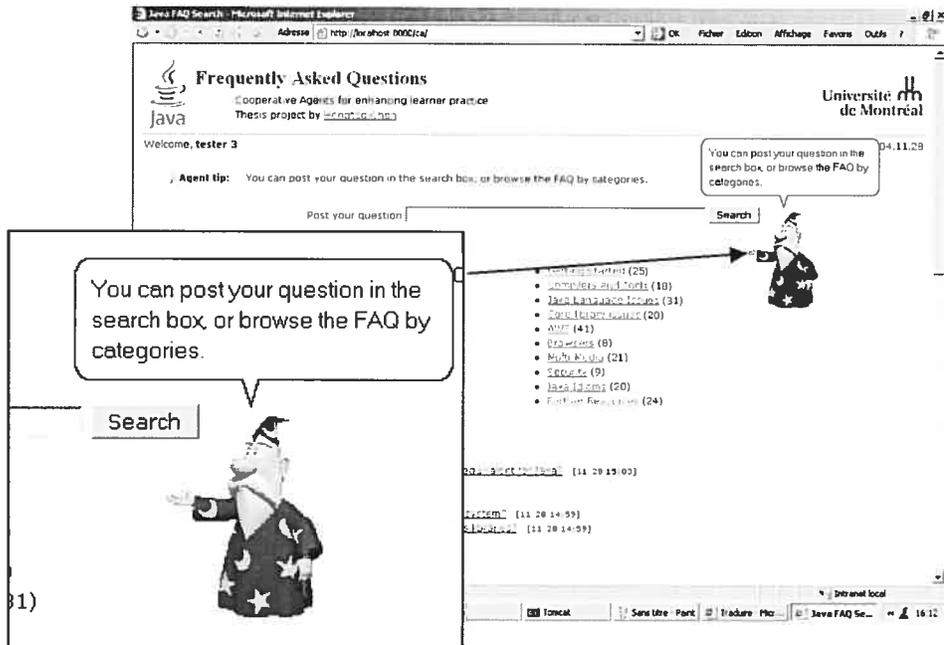


Figure 7-20: Agent suggests the learner to use the search box.

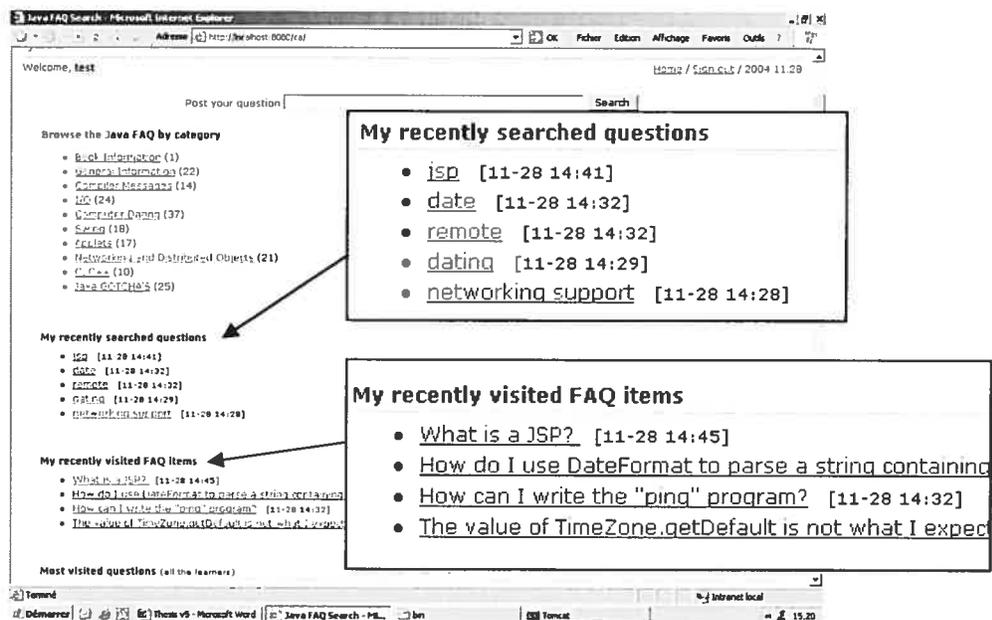


Figure 7-21: Learner's Start Page

Below the category list, it's the personal question list and item list. After several times visiting, every learner could have their own interesting list and question list (Figure 7-21). So the learner can have his/her own learning route instead of navigating inside the categories. Because they can quickly select his/her interesting questions or items in these two lists, the more the learner uses the application the more the learner can complete his/her question or learning topic.

Meanwhile, the personal agent also pops up some messages to suggest the learner according to his/her behaviour history. For instance, recommend the learner to visit his/her interested category (Figure 7-22).

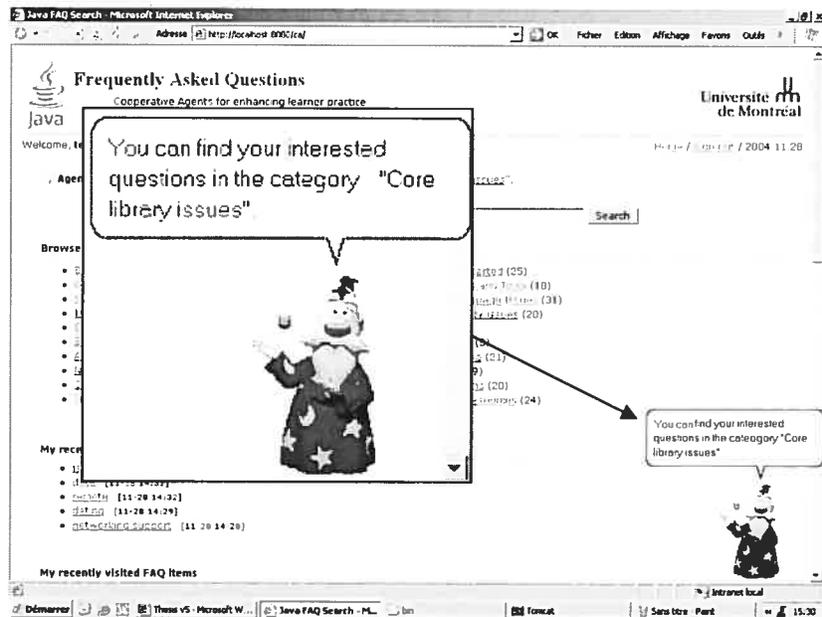


Figure 7-22: Recommend the learner

By these features, every learner can have his/her personalized learning process consistent with his/her interesting and learning history. Learner can quick find out the questions that he/she asked before to review the answer; can go to the FAQ item he/she visited last time directly. Therefore, we can say that the learning process is organized by the learner himself instead of the learning system.

Scenario 2: Searching

In our application, agents work at the background to help the learner to get the convenient results when learner searches his/her question. For instance, Motivation agent could suggest the learner to check the specific category where the most searching results come from (Figure 7-23).

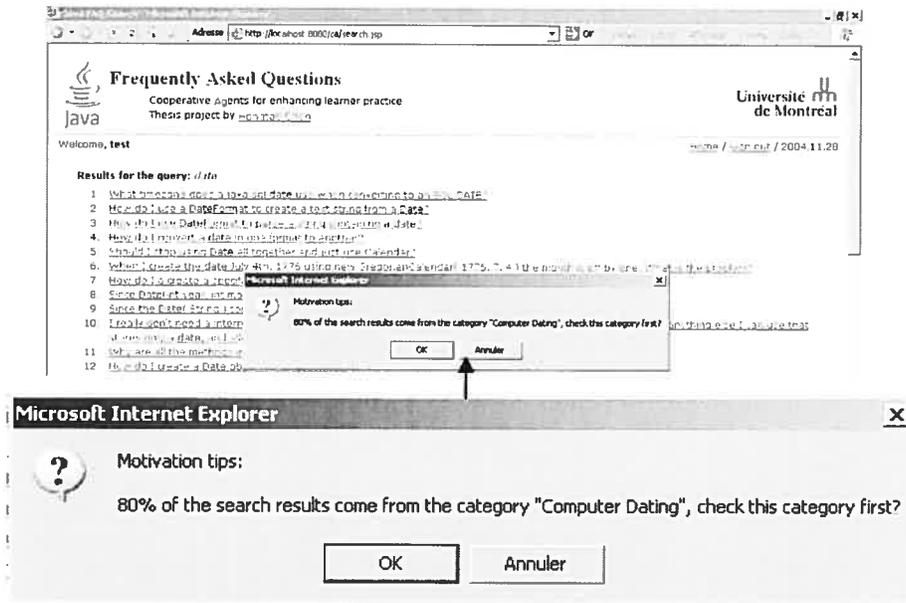


Figure 7-23: Motivation Tips

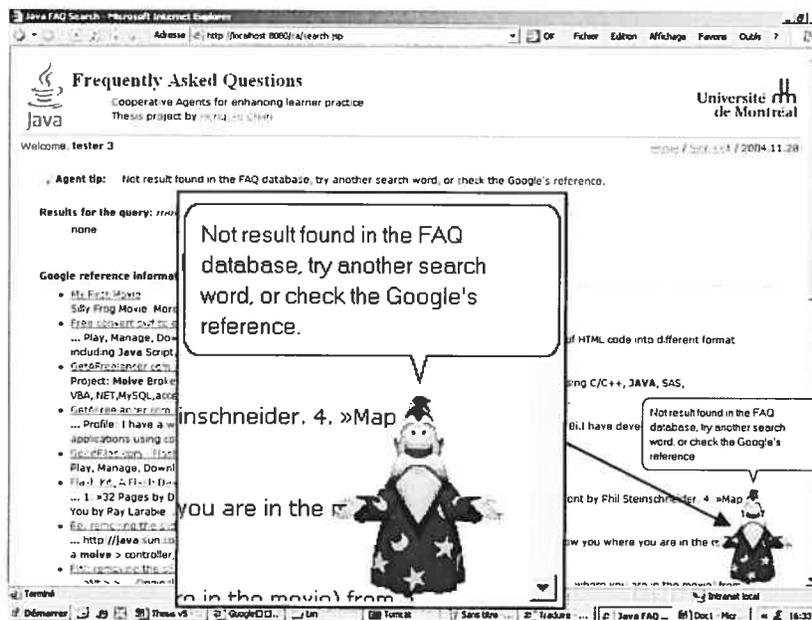


Figure 7-24: Searching without result

On the other hand, a Google reference is always included in each search page. Agent can suggest the learner to refer to the Google reference if the search engine cannot find the suitable result in the FAQ database (Figure 7-24). Essentially, the Google reference makes the application’s search capacity almost infinite.

In additional, the application also provides a valuable feature that the learner can search any text displayed in the application. From the mouse right click popup menu, learner can find the definition of the selected text; or search these words in Google (Figure 7-25).

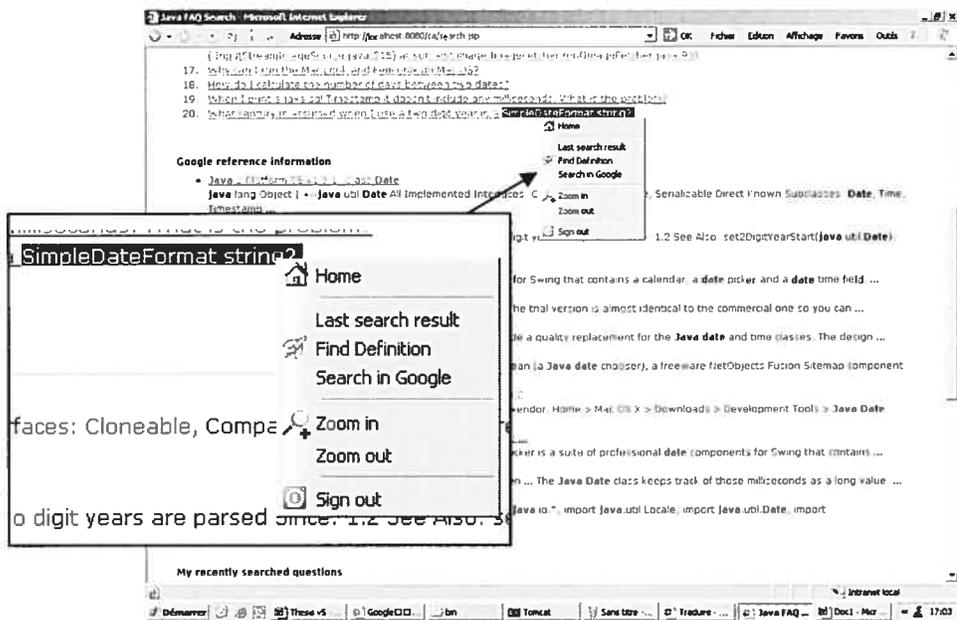


Figure 7-25: Search selected text in the web page

Scenario 3: Sharing recommendation

The following two figures illustrate the difference before and after that a learner recommends a reference page. In the first figure, the reference page “Servlet Tutorial: Java Server Pages (JSP) 1.0” shows in the 4th place of the list. When a learner read this page and recommends it to another learner, this page becomes the first one in that list (Figure 7-27). Other learners can easily get benefit by the rating stars after the topic title.

It's one of the important features that the application system supports learners to share their learning experience each other in a cooperative environment.

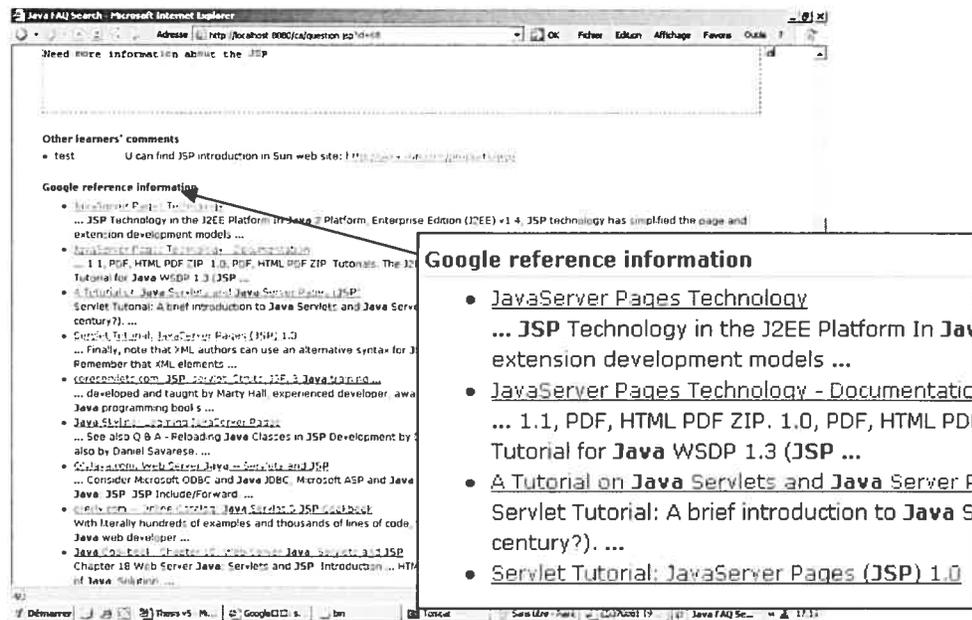


Figure 7-26: Original List

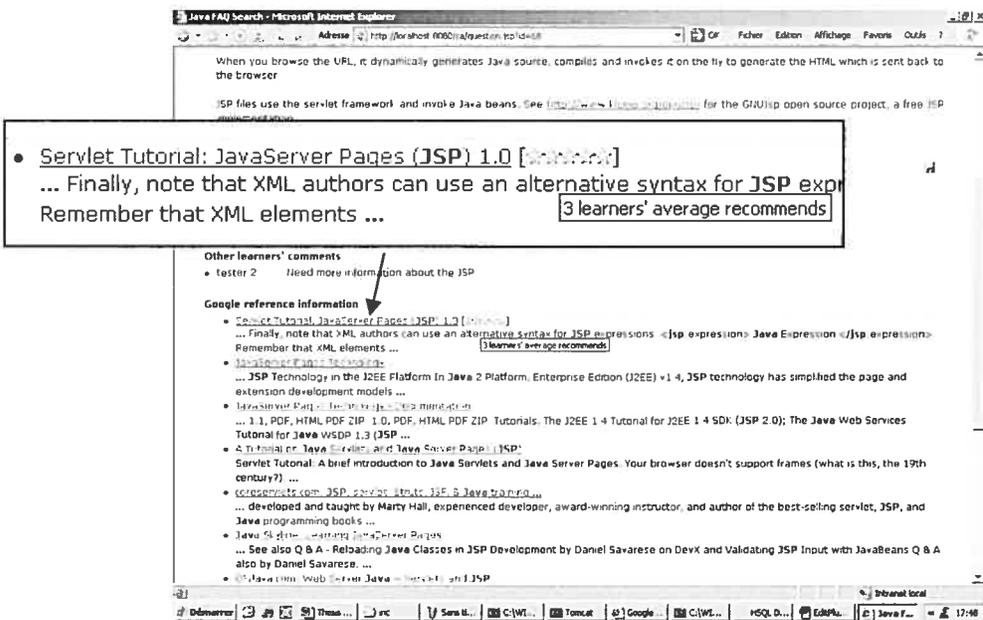


Figure 7-27: After learner's recommendation

Scenario 4: Post personal comment in the searching result

Posting comment on the learning topic is another feature in the application that supports learners exchange their learning experiences in the cooperative environment.

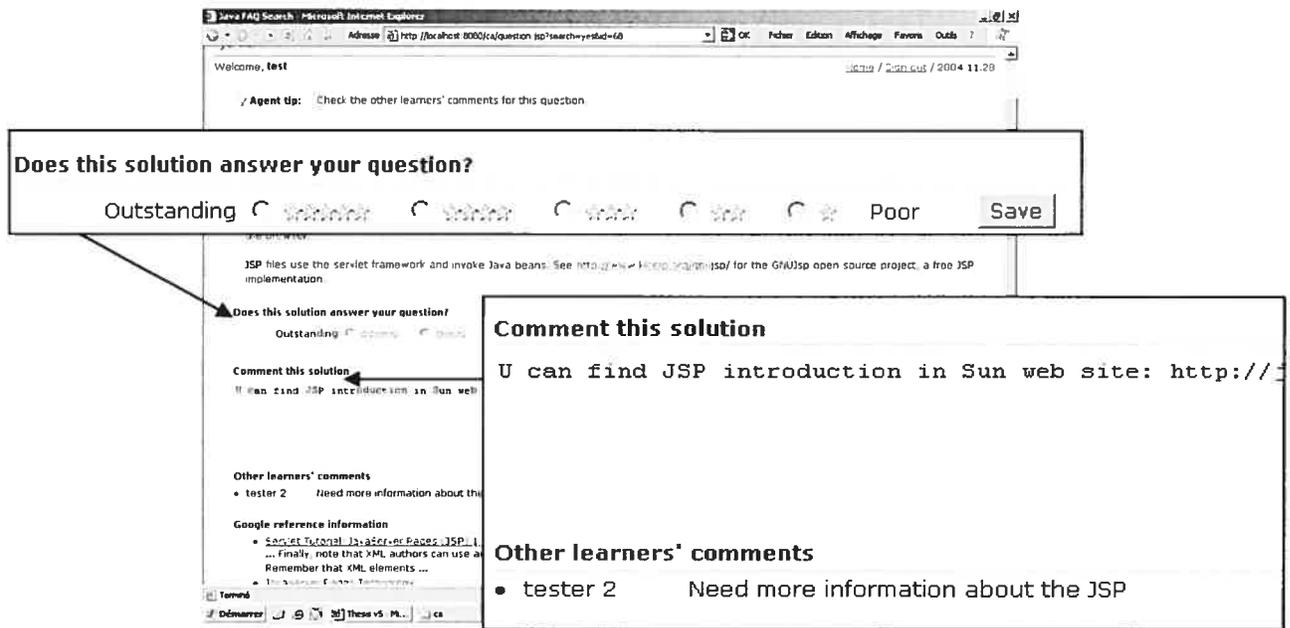


Figure 7-28: Comment the topic

Learners can ask their questions and exchange their options about the topic by posting comments on it. This feature makes the application as a learning platform rather than a FAQ searching system.

7.6 Summary

In this chapter, we presented the implementation of our cooperative learning environment. We began with the run-time environment and development tools. After the introduction of the class and scenario diagrams, we end this chapter with the description of the user interfaces in the system.

Conclusion

The goal of this thesis is to study architectures that are designed to function in online learning environment to let multiple agents enhance the learning practice. We present a framework of the cooperative learning environment (CLE) which aims to the goal above by providing a personalized learning support. We also present an initial version of Java FAQ application – an agent based online learning system designed to help enhancing the learner's practice in a cooperative learning environment by providing personalize learning support.

8.1 Conclusion

Intelligent agents can be employed to shift online learning paradigms away from a traditional learning environment to concentrate instead on a learner's individual needs. An online learning environment with intelligent agents can help move learners toward an apprenticeship, or learn-while-doing approach. The agents system demonstrates that agent technology can successfully work in place of a human tutor to give immediate responses in a personal learning environment and also help individual learners' communication in a CLE.

We found that using intelligent agents in our CLE showed a very positive association with a higher satisfaction rate. Some of observed outcomes including: With the agent's participation, there was a dramatic increase of learner's satisfaction, and the majority of the learners expressed positive attitudes toward using the agents, specifically as a tool

that helped motivate them to complete the learning session. Agents' roles in online learning environment including:

Agents as a Motivational Tool

The results from the thesis study supported the notion that intelligent agents in the form of agents can be used as a motivational tool. The results from the correlation analysis also indicated a positive correlation between the number of times the learners used the agents and the number of learning topics completed by the participants. For instance, because the application always displays the most recent questions and topics of the individual learner, the more the learner uses the application the more the learner can complete his/her question and learning topic.

Results from the survey analysis on motivation indicated that features of agents such as personalized topic selection, relative resource presentation, other learners' commentary and recommendations helped motivate the participants to complete the topic and discover the advanced information. These features are of positive benefit to the FAQ study when agents are present in the online learning environment.

Specifically, the commentary and recommendation that other learners provided through the CLE helped motivate the learners to stay focused on the topic and extend the knowledge around it. Explicit directions on where to find out the relative resource about the topic were found by learners to be useful in feature. Providing explicit help to learners improved the motivation rate of the subsequent session. Our conclusion is that agents can be a strong motivational tool.

Agents as a Tutor

In an online learning system, a very high number of learners had positive attitudes toward the use of agents as a learning tutor. A likely reason is that agents provided personalized support to them when they needed it. Personalized support, including customized learning topic selection, presenting learners with learning history list, helped the learners to quickly solve their problems. Other than reporting the searching history, personalized support also provided other assistance about where to find information in the learning materials and where to seek further help. In these cases, agents helped learners to reduce the time required to find answers to their problems. Anytime, personalized support also facilitated self-paced learning. Some learners prefer to move at their own style; it is indeed possible that the agent system supported such learners better than class-oriented exercises.

Agents as a Human-to-Human Interaction Facilitator

In a personalized learning support system, agents increase the need for interaction with other. This finding suggests that our agents provided another mechanism for stimulating discussions and learner-to-learner interactions, not the converse. Agent's notification about the topic discussion would develop the learner to spend more time into the discussion and observably increase the communication between learners.

In conclusion, the results from the study indicated that use of intelligent agents is significantly associated with learner progress. The agent system demonstrated that agent technology can supplement a human tutor to give personalized instruction and support human-to-human interactions.

8.2 Future work

In spite of the promises of agent-based online learning environments, however, there still have some constraints. First, agent-based system requires intelligence and adaptability in

order to substantiate their potential. Current technology is still limited to fully construct artificially “intelligent” online learning system. Second, even without developing artificially intelligent online learning system, designing and developing online learning system is demanding technologically. While there are some ready-made agents available (e.g., Microsoft Agent characters), it is usually necessary for researchers to develop their own system for their particular research questions. Lastly, research on online learning and agent-based system needs to be interdisciplinary in nature involving instructional design, cognitive psychology, human computer interaction, artificial intelligence, and communication. While this is advantageous in promoting more ecologically valid research, it is difficult to coordinate and conduct collaborative research drawing from such diverse fields.

Even with these constraints, the use of agent-based learning companions has significant promise in shaping a new paradigm in computer-based learning. Agent-based online learning system can serve as lifetime learning partners given that learning is a lifelong process. New technologies, such as mobile computing and virtual reality will also likely have a place for these kinds of system whenever social interaction and supports are needed to assist learner in reaching the intended outcomes.

As described above, agent-based learning environments are evolving with technological advancement and continuous research, the findings from which could produce more constituents and extend the design of agent-based learning to further sophistication.

There are still many important questions which remain unanswered. The agents system is designed to enhance the learner’s practice through providing personalized learning support in an online cooperative learning environment. In our current study, we were



unable to precisely measure the effect which agents encourage to learner, such as the approval comparison before and after using the agents.

Future work will verify the performance of agent-based online learning system by extended experiments specifying the optimal model of personalized learning support, because few researches has tried to specify the advantages of agent-based personalized learning support. In addition, we expect that application will show further improved result since the increased number of occurrence terms in the query may affect the personal scheme positively.

Bibliography

- [Adams 2001] Julie A. Adams (2001) "Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence. (Review)" In AI Magazine, Summer, 2001 http://www.findarticles.com/cf_dls/m2483/2_22/76698535/print.jhtml
- [Alboaie *et al.* 2002] Sinica ALBOAIE, Sabin-Corneliu BURAGA, Lenuta ALBOAIE (2002) "An XML-based Object-Oriented Infrastructure for Developing Software Agents" Scientific Annals of Cuza University, Volume 12, p 109-134
- [Alonso *et al.* 2001] Eduardo Alonso, Mark d'Inverno, Daniel Kudenko, Michael Luck, and Jason Noble (2001) "Learning in Multi-Agent Systems" Knowledge Engineering Review 16(3):pp. 277-284.
- [Andrade *et al.* 2001] Adja F. de Andrade, Patrícia A. Jaques, João Luiz Jung, Rafael H. Bordini, Rosa M. Vicari (2001) "A Computational Model of Distance Learning Based on Vygotsky's Socio-Cultural Approach" Proceedings of the MABLE Workshop (Antonio, Texas, May 19-23). X International Conference on Artificial Intelligence on Education.
- [Arnold 2000] Josie Arnold (2000) "eTeaching and eLearning: Enhancing teaching and learning using the new technologies" http://www.ld.swin.edu.au/staff_dev/entered/assets/images/eTeaching_eLearning.pdf
- [Baylor 1999] Baylor, A.L. (1999) "Intelligent agents for education." Paper presented at the American Educational Research Association, Montreal, Canada 1999.
- [Billard and Hayes 1997] Billard, A., & Hayes, G. (1997). "Learning to communicate through imitation in autonomous robots" In Proceedings of The Seventh International Conference on Artificial Neural Networks, pp. 763-768 Lausanne, Switzerland.
- [Billsus *et al.* 2000] Daniel Billsus and Michael J. Pazzani and James Chen (2000) "A learning agent for wireless news access" Intelligent User Interfaces Pages 33-36
- [Blackboard 2000] Blackboard Inc. (2002) "Educational Benefits of Online Learning" on www.blackboard.com
- [Bransford, *et al.* 2002] John D. Bransford, Ann L. Brown, and Rodney R. Cocking, Editors (2002) "How People Learn" Committee on Developments in the Science of Learning, National Research Council

- [Brown *et al.* 1998] Brown S. M., Santos Jr. E., Banks S. B., & M. E. Oxley (1998) "Using Explicit Requirements and Metrics for Interface Agents User Model Correction" Proc. Second International Conference on Autonomous Agents, Minneapolis, St. Paul, MN. 1-7.
- [Buraga 2003] Sabin-Corneliu Buraga (2003) "Developing Agent-Oriented e-Learning Systems" Proceedings of The 14th International Conference on Control Systems And Computer Science – vol.II, I.Dumitrache and C.Buiu (eds.), Politehnica Press, Bucharest
- [Chan 1990] Chan, T. and Baskin, A.B. (1990) "Learning companion systems" In C. Frasson and G. Gauthier (eds) Intelligent Tutoring Systems: At the Crossroads of Artificial Intelligence and Education. pp. 6--33. Norwood, NJ: Ablex.
- [Dahlstrom and Wiewiora 2002] Dana Dahlstrom, Eric Wiewiora (2002) "Imitation in Reinforcement Learning" <http://www.cs.ucsd.edu/users/ddahlstr/cse254/proposal.pdf>
- [Davies and Sklar 2003] Mathew Davies and Elizabeth Sklar (2003) "Modeling Human Learning as a Cooperative Multi Agent Interaction" AAMAS-03 The International Conference on Autonomous Agents and Multiagent Systems July 14, 2003 Melbourne, Australia
- [Dumas 2003] Alexandre Dumas (2003) <http://www.cde.ca.gov/iasa/cooplrg2.html>
- [Ferber 1999] Jacques Ferber (1999) "Multi-Agent System: An Introduction to Distributed Artificial Intelligence" ISBN 0-201-36048-9
- [Franklin and Graesser 1996] Stan Franklin, Art Graesser (1996) "Is it an Agent, or just a Program?: A Taxonomy for Autonomous Agents" Proceedings of the Third International Workshop on Agent Theories, Architectures, and Languages, Springer-Verlag.
- [Frasson *et al.* 1996] Claude Frasson, Thierry Mengelle, Esma Aïmeur, Guy Gouardères (1996) "An Actor-based Architecture for Intelligent Tutoring Systems" Intelligent Tutoring Systems, Springer Verlag, Berlin, p. 57
- [Frasson *et al.* 1997] Claude Frasson, Thierry Mengelle, Esma Aïmeur. (1997) "Using Pedagogical Agents in a Multi-Strategic Intelligent Tutoring System" AI-ED97: Eighth World Conference on Artificial Intelligence in Education-Workshop V: Pedagogical Agents, 8 1997. Proceedings... Kobe: Japan, 1997.
- [Genesereth and Ketchpel 1994] M.R. Genesereth and S.P. Ketchpel (1994) "Software Agents", Communications of the ACM, vol. 37, no. 7, pp. 48-53, July 1994.

- [Gilbert 1997] Don Gilbert (1997) "Intelligent Agents: The Right Information at the Right Time" IBM Report, IBM Corporation, Research Triangle Park, NC, USA (May 1997): <http://www.networking.ibm.com/iag/iaghome.html>
- [Hawryszkiewicz 2003] I.T. Hawryszkiewicz (2003) "Agent Support For Personalized Learning Services" Proceedings of the The 3rd IEEE International Conference on Advanced Learning Technologies (ICALT'03)
- [Huang 2003] William Huang, MD (2003) Providing Difficult Feedback: TIPS for the Problem Learner
- [Huang and Edwards 2003] Hsuanchao Huang & Reuben Edwards (2003) Paradigm of instructional agent system PNet 2003 Conference.
- [IBM] "The IBM Agent" <http://activist.gpl.ibm.com:81/WhitePaper/ptc2.htm>
- [Jamroga 2003] Wojciech Jamroga (2003) "Confidence Measure for a Learning Agent" Proceedings of Eighth Scandinavian Conference on Artificial Intelligence (SCAI'03), B. Tessem et al. (eds.), November 2003, Bergen, Norway, Frontiers in Artificial Intelligence and Applications, IOS Press, 95-106.
- [Joseph and Kawamura 2001] Sam Joseph, Takahiro Kawamura (2001) "Why Autonomy Makes the Agent" In Agent Engineering Eds. Liu, J, Zhong, et al. World Scientific Publishing (2001)
- [Kameda and Nakanishi 2003] Tatsuya Kameda and Daisuke Nakanishi (2002) "Cost-benefit analysis of social/cultural learning in a non-stationary uncertain environment" Evolution and Human Behaviour, 23 (2002) pp. 373-393
- [Labidi and Silva 2000] Sofiane Labidi and Jeane Silva Ferreira (2000) "Technology-Assisted Instruction Applied to Cooperative Learning: the SHIECC project" To appear in the International Journal of Continuous Engineering and Life-Long Learning. Special Issues on Intelligent Agents for Education and Training System. 2000.
- [Lesser 1999] Victor R. Lesser, "Cooperative Multiagent Systems: A Personal View of the State of the Art" Knowledge and Data Engineering, Vol. 11, No. 1 (1999), 133--142.
- [Looi 2001] C.K. Looi (2001) "Enhancing learning ecology on the Internet" Journal of Computer Assisted Learning 17, no. 1 (2001): 13-20.
- [Lucia Maria and Rosa Maria 1998] Martins Giraffa Lucia Maria, Viccari Rosa Maria (1998) "The Use of Agents Techniques on Intelligent Tutoring Systems" RIBIE 98, IV Congresso da Rede Iberoamericana de Informática Educativa

- [Mehrddad 2003] Mehrddad 2003 <http://www.ee.umd.edu/serts/bib/thesis/mehrdad/>
- [Mengell and Frasson 1996] Thierry Mengelle and Claude Frasson (1996) "A multi-agent architecture for an ITS with multiple strategies" CALISCE 1996: 96-104
- [Nicholas 2000] R. Jennings (2000) "On agent-based software engineering" Artificial Intelligence, (117): 277--296, 2000.
- [Odell 2001] James Odell 2001 "Autonomous Agent" <http://c2.com/cgi/wiki?JamesOdell>
- [Odell 2002] James Odell (2002) "Objects and Agents Compared" JOURNAL OF OBJECT TECHNOLOGY Vol. 1, No. 1, May-June 2002 page 41-53
- [Ojo, et al. 2002] A. K. Ojo, A.M Rahman, and H.O.D Longe. (2002) "Agent Based User-Modeling" ICTEI'2002 Workshop, Ibadan, Nigeria, August, 2002
- [Olguín *et al.* 2000] Carlos José M. Olguín, Armando Luiz N. Delgado & Ivan Luiz M. Ricarte (2000) "An Agent Infrastructure to set Collaborative Environments" Educational Technology & Society Vol.3, No.3.
- [Opera 2002] Mihaela Oprea (2002) "An Adaptive Negotiation Model for Agent-Based Electronic Commerce" Studies in Informatics and Control, Vol.11, No.3, September 2002 Pp 271-279
- [Paletta and Rome 2000] Lucas Paletta, Erich Rome (2000) "Reinforcement Learning of Object Detection Strategies" SIRS 2000, 8th International Symposium on Intelligent Robotic Systems, The University of Reading, England, July 18-20, 2000.
- [Palthepeu 1991] Palthepeu, S., Greer, J., and McCalla, G. (1991) "Learning by Teaching" The Proceedings of the International Conference on the Learning Sciences, AACE.
- [Parunak and Van Dyke 1997] Parunak, H. Van Dyke, "Go to the Ant: Engineering Principles from Natural Agent Systems" Annals of Operations Research, 75, 1997, pp.69-101.
- [Peeters 2003] Maarten Peeters (2003) "A Study of Reinforcement Learning Techniques for Cooperative Multi-Agent Systems" PhD thesis
- [Pattie 1995] Maes, Pattie (1995), "Artificial Life Meets Entertainment: Life like Autonomous Agents" Communications of the ACM, 38, 11, 108-114
- [Pinar 1996] Pinar Öztürk (1996) Distributed Artificial Intelligence and Intelligent Agents <http://www.idi.ntnu.no/~agent/>

- [Plaza and Ontañon 2003] Enric Plaza, Santiago Ontañon (2003) "Cooperative Multiagent Learning" Adaptive Agents and Multi-Agent Systems, Lecture Notes on Artificial Intelligence 2636, Springer Verlag
- [Price and Boutilier 1999] Bob Price, Craig Boutilier (1999) "Implicit imitation in multiagent reinforcement learning" Proceedings of the Sixteenth International Conference on Machine Learning, pages 325–334, 1999. Morgan Kaufmann, San Francisco, CA.
- [Prodromidis and Stolfo 1999] Andreas L. Prodromidis and Salvatore Stolfo (1999) "Agent-Based Distributed Learning Applied to Fraud Detection" Sixteenth National Conference on Artificial Intelligence
- [Razek *et al.* 2002] Razek M., Frasson, C., Kaltenbach M. (2002) "A Confidence Agent: Toward More Effective Intelligent Distance Learning Environments" ICMLA'02: Las Vegas, USA, June 24-27, 2002.
- [Rosenschein and Kaelbling 1995] S.J. Rosenschein and L.P. Kaelbling, (1995) "A Situated View of Representation and Control", Artificial Intelligence 73(1-2), pp.149-173
- [Rosić *et al.* 2002] M. Rosić, S. Stankov, V. Glavinia (2002) "Personal Agents in Distance Education Systems" INES 2002, 6th International Conference on Intelligent Engineering Systems 2002, May 26-28, 2002, Opatija, Croatia
- [Russel, and Norvig 1995] S. Russel, and P. Norvig, (1995) "Artificial Intelligence: A Modern Approach", Prentice Hall.
- [Salton and Guckley 1990] Salton, G., Buckley, C. (1990) Improving retrieval performance by relevance feedback. JASIS 41, 1990, pp. 288-297.
- [Santos and Touzet 2000] Juan Miguel Santos, Claude Touzet (2000) "Dynamic Update of the Reinforcement Function during Learning" Connection Science, Special issue on Adaptive Robots.
- [Seiker 1994] Seiker, T. (1994). "Coach: A teaching agent that learns" Communications of the ACM, 37(7), 1992-1999.
- [Sen and Sekaran 1999] Sandip Sen, Mahendra Sekaran (1999) "Individual Learning of coordination knowledge" Journal of Experimental & Theoretical Artificial Intelligence, 10:3, pp. 333-356.
- [Shimizu *et al.* 2003] Noritada Shimizu, Jun'ichi Nakamura, Nofumi Yoshida, Takashi Hattori, Tatsuya Hagino (2003) "Personalization of Materials for Learning on Demand

Using RDF” The Twelfth International World Wide Web Conference 20-24 May 2003, Budapest, UNGARY

[Stone and Veloso 2000] Peter Stone, Manuela Veloso, (2000) “Multiagent Systems: A Survey from a Machine Learning Perspective” *Autonomous Robots*, 8(3):345-383

[Sugawara and Lesser 1998] Toshiharu Sugawara, Victor Lesser. (1998) “Learning to improve coordinated actions in cooperative distributed problem solving environments” *Machine Learning*, 33(2/3):129-153.

[Vidal and Buhler 2001] Jos’e M. Vidal, Paul Buhler (2001) “A Generic Agent Architecture for Multiagent Systems” USC CSCE TR-2002-011

[Vizcaino and Boulay 2002] Aurora Vizcaino, Benedict du Boulay (2002) “Using a Simulated Student to Repair Difficulties in Collaborative Learning”

www.cogs.susx.ac.uk/users/bend/papers/icce2002.pdf

[Webber *et al.* 2002] Carine Webber, Sylvie Pesty, Nicolas Balacheff (2002) “A multi-agent and emergent approach to learner modelling” ECAI 2002 - Proceedings of the 15th European Conference on Artificial Intelligence. F. van Harmelen (ed.), IOS Press, Amsterdam, 2002. Pp.98–102.

[Weiss and Dillenbourg 1999] Gerhard Weiss, Pierre Dillenbourg (1999) “What is ‘multi’ in multi-agent learning” P. Dillenbourg (Ed.) *Collaborative Learning: Cognitive and Computational Approaches*. Amsterdam : Pergamon/Elsevier Science

[Wentling *et al.* 2000] Tim L. Wentling, Consuelo Waight, Danielle Strazzo, Jennie File, Jason La Fleur, Alaina Kanfer (2000) “The Future of e-Learning: A Corporate and an Academic Perspective” University of Illinois, Urbana-Champaign, <http://learning.ncsa.uiuc.edu/papers/learnfut.pdf>

[Wooldridge and Jennings 1992] Wooldridge, M.J. Jennings, N.R. (1992) “Agent Theories, Architectures, and Languages: A Survey.” In M.J. Wooldridge and N.R. Jennings (Eds.)

[Wooldridge and Jennings 1995] Michael Wooldridge and Nick Jennings (1995) “Intelligent Agents: Theory and Practice” *Knowledge Engineering Review* Volume 10 No 2, June 1995. (c) Cambridge University Press, 1995.

[Zhang *et al.* 2003] G. Zhang, Z. Cheng, A. He, T. Huang. (2003) “A WWW-based Learner’s Learning Motivation Detecting System” *Proceedings of the Conference on Knowledge Economy and Development of Science and Technology*



Handwritten text at the bottom right of the page, possibly a signature or date.