129160

COMET-KO: A CONCEPT MAPPING ENVIRONMENT FOR KNOWLEDGE ORGANIZATION

A THESIS
SUBMITTED TO THE DEPARTMENT OF COMPUTER ENGINEERING
AND THE INSTITUTE OF ENGINEERING AND SCIENCE
OF BILKENT UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE

129160

By
Tayfun Küçükyılmaz
September, 2002

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

[illegible]

Asst. Prof. Dr. David Davenport (Advisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

[illegible]

Assoc. Prof. Dr. Özgür Ulusoy

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

[illegible]

Asst. Prof. Dr. Uğur Doğrusöz

Approved for the Institute of Engineering and Science.

[illegible]

Prof. Dr. Mehmet B. Baray
Director of the Institute

ABSTRACT

Comet-KO: A CONCEPT MAPPING ENVIRONMENT FOR KNOWLEDGE ORGANIZATION

Tayfun Küçükyılmaz
M.S. in Computer Engineering Supervisor: Asst. Prof. David Davenport
September 2002

This thesis presents Comet-KO, a software tool for knowledge construction and organization, which takes its basis from instructivist and constructivist learning theories. Comet-KO aids users to collect, create and represent information, visualize relationships between concepts, and collaborate during the knowledge construction process. The representation of knowledge in Comet-KO is based on semantic networking techniques, which proved to be more powerful than conventional text based note-taking methodologies. In addition to formal semantic networking language, grouping and hypergraph drawing is also supported by Comet-KO, which brings a new perspective on visualization of knowledge construction.

Keywords: Instructivism, Constructivism, Concept Maps, Semantic Networking, Knowledge Organization,

ÖZET

Comet-KO: BİLGİ DÜZENLEMESİ İÇİN BİR ANLAMSAL AĞ ORTAMI

Tayfun Küçükyılmaz
Bilgisayar Mühendisliği Yüksek Lisans Tez Yöneticisi:
Yard. Doç. Dr. David Davenport
Eylül, 2003

Bu tez temellerini instrüktivist ve konstrüktivist eğitim teorilerinden alan bir bilgi inşaa ve düzenleme olan Comet-KO`u sunmaktadır. Comet-KO, bilgi inşaası ve düzenlemesi sürecinde kullanıcıya bilgiyi toplama, yaratma ve sunmada, bu bilgileri ilşkilendirmede ve bu süreçte diğer kullanıcılarla etkileşmede yardımcı olmayı amaçlamaktadır. Comet-KO bilginin ifadesini yazıya dayalı klasik bilgi sunum methodları yerine, bilgi inşaasında bu tekniklerden çok daha üstün olan mantıksal ağ teknikleri ile yapmaktadır. Bilinen anlamsal ağ tekniklerini gruplama ve hipergraf çizim gücü ile geliştiren Comet-Ko, bilginin sunumu konusunda en kapsamlı anlamsal ağ geliştirme araclarından biri olmaya adaydır.

Anahtar sözcükler: İnstrüktivizm, Konstrüktivizm, Anlamsal Ağlar, Bilgi Düzenlemesi

Contents

List of Figures

Chapter 1

Introduction

Today, in the knowledge age, the need for up to date information is paramount. Society demands individuals with self-learning, cooperative working and creative thinking abilities. These abilities are not innate; people need to be trained in order to develop them.

Unfortunately, current instructivist education seems to have little correlation with these knowledge age skills. In contrast, Social Constructivism, an emerging pedagogical theory, places self-regulated learning at the center of an educational system. As a learner-centered theory, social constructivism claims that learning should be based on experiencing phenomenon and brainstorming. By experiencing, individuals not only gather information, but also relate it with past knowledge, criticize it, and even develop it further. In addition, collaboration is an integral part of social constructivist education theory, according to which, learning is an individual process directed by interactions between learners in a social community. Learners, as social partners, learn how to solve difficult problems by assisting each other and working collectively.

CHAPTER 1. INTRODUCTION

This thesis presents the design and implementation of Comet-KO (Concept Mapping Environment for Knowledge Organization), a software based on social constructivist theory. Comet-KO is as a collaborative environment where users visualize and organize knowledge. Concept maps, or semantic networks, model the perception mechanism of the human brain, and thus, offer an excellent technique for visually representing the ideas. Concept maps are graph-like diagrams where nodes represent units of information (concepts), and edges represent the relation between these concepts. Comet-KO, as a knowledge organization environment with computer-mediated collaboration features aims to provide a tool to facilitate learning in both teacher and learner centered classrooms.

The organization of the thesis is as follows: In Chapter 2, the major educational theories, instructivism and constructivism, are presented, the social effects on education are identified and social constructivism is discussed. Chapter 3 presents a brief overview of educational technologies. Powerful attributes of computer-mediated studies are identified as various supporting features of our work. Chapter 4 introduces concept mapping, the knowledge representation tool used in the thesis work. The features of a concept map are presented and possible improvements on formal concept mapping techniques are discussed. A number of existing software systems which employ concept mapping techniques are examined. Chapter 5 explores the possible uses of the proposed design. Finally, chapter 6 presents the features, design, architecture, and implementation details of Comet-KO. The thesis concludes in chapter 7, with a summary of the work done and suggestions for future directions.

Chapter 2

Education and Learning

Before going into the detail of work done in this thesis, it is essential to identify the motivations behind it. As an educational tool, Comet-KO should take its basis from social psychology and pedagogy. In this chapter, different educational theories are examined in detail.

2.1 Education as a Means of Learning

The social sciences can be defined as theory-poor sciences [1], especially when it comes to pedagogy (Educational sciences). The reason is that, no proved fact can be obtained in the field of pedagogy, since experiments done in the field depend on human interactions and cognition, which can be considered as uncontrollable attributes. Even if all attributes of the experiment is selected objectively, no one can defend that the subject groups are identical. Two individuals coming from the same social and cultural basis cannot be perceived as the same.

As a result of this many totally different ideas arise on human cognition, ideal education system, learning system and even how learning process is handled by human mind. Some of these ideas refer to social effects while some to individualism. Some theories support experts as tutors while some support experts as guides or coaches. Today, with the emergence of new technologies, active-learning strategies, an education based on learning with experience find more support. The active-learning strategies claim that meaning making process is based on activities. These activities are perceived, associated and constructed individually in human brain.

2.2 Human Mind and Learning Process

How do we learn our knowledge and skills? This is the one and only question behind cognitive psychology. How does the human brain operate? All learning processes and as a result, all educational theories are centered on this simple question, the answer to which is far from straightforward. Psychologists are divided into several camps about this issue. Behaviorists and cognitive psychologists are the most referred ones.

The behaviorists believe that the learning process is defined as a change in individual's behavior. A trainer can change this behavior via a set of activities independent of the individual. The role of the trainer here is measuring the responses supplied by the trainee to these activities, and making adjustments, which change target learners' behavior in order to get most performance.

The cognitivists perceive the subject from a very different angle. They believe that the learning takes place as brain cells process the outside information and also that the brain cells' connectivity differs from person to person. This idea accepts that learning is uniquely different for each individual. Here, the role of the trainer is to select a set of activities that best matches the individual and his needs.

One other highly regarded theory is Gardner's "Multiple Intelligences" [ayse]. According to this theory, brain is composed of seven distinguishable parts, which have different capabilities. Examples of these regions are logic, musical, bodily, and etc. Each part determines the intelligence (in other words, skill) of an individual at the specified subject. Each region may function at different proportions at each individual, so each individual may have different intellectual abilities. According to Gardner's theory an individual's success in a field depends on the brain wiring that is inherited. For a successful education, the trainer's role here is to determine a set of activities that covers maximum range of intelligences.

There are other less widely supported ideas including:

- Human beings construct knowledge by actively selecting information and connecting it with past knowledge and experiences.
- A concept is stored in a particular part of the brain around which the related facts, both new and old, are connected. Unrelated knowledge means that it is unlinked, thus more vulnerable to forgetting [3].

With such a wide range of different and opposed ideas, the pedagogical theories also have different perspectives.

2.3 Educational Theories

As the answers to the question "How does the human brain operate?" vary considerably, so do the educational theories. They may be broadly divided into two main categories: Instructivist and constructivist point of view; also known as teacher-centered and student-centered theories respectively. As mentioned earlier in this chapter, with the emergence of computer into the educational field, the constructivist theories gain popularity. At the same time, collaborative classroom also emerges with the growth of hypermedia and the Internet. As a result, there is a misconception that the instructivism is opposed to classroom collaboration. In fact, this is not the case, though most of the ongoing work on collaborative education is done with a constructivist point of view. The purpose here is not to degrade instructivism or promote constructivism. Both views have some strong points. One of the purposes of this research is to find a way to combine these two ideas.

2.3.1 Instructivism

Instructivism, which is also referred to as objectivism, is a pedagogic theory that is directed by a behaviorist point of view. Therefore, it is also classified as a "Teacher-centered" or an "Instruction-based" theory. Today, the instructivist theory is applied in most of the educational associations, save for some colleges, test classes in primary schools and graduate education. The teacher instructs students and evaluates them with quizzes, exams, multiple-choice tests, etc. The classroom is the place where the authority belongs to the teacher in instructivist settings. Students in the classroom are subjects to be filled with appropriate information.

In fact, the instructivist theory is as old as mankind. It is driven by " The heritage of information". If there are experts of a subject, it is not necessary to re-invent the wheel. These experts can simply guide others. By time, this expert becomes a classroom teacher while trainees become the class. As behavioral theory suggests, according to the instructivist perspective, given the appropriate tools, an expert can alter anyone's behavior to what is expected [3]. Tools here refer to the instructions presented by the teacher. The role of the teacher in the classroom is to give these instructions to trainees and make sure that trainees get the instructions. The teacher makes evaluation process by observing the responses of the trainees. Today, exams, tests, and many methods are used to evaluate the direct responses of trainees. However, the instructivism simply neglects the possible subjectivity of this process. While instructivism explains this as activities that learners respond most positively [3], I have no doubt that appropriate means the expectations and criterion of the society, not that the trainee learns what is needed.

Clearly, instructivism is driven by the social effects within the community. However, it lacks the individual effects on the learning process. Teachers evaluate the *responses* of the whole trainee group while there is a strong possibility that individuals do not have the exact same view or capacities for the instructions. Thus, we can say that instructional education is *equa*l with respect to the society while it is *unequal* with respect to trainees. The reason is simple. Training enough experts for trainees is impossible. Today millions of people enter the University Qualification Exam (OSS) in Turkey, while only a couple of thousands are accepted. The whole world suffers from the same situation. It is commented that [4] in America almost 20 million 12th grade students do not even have the basic reading writing skills. Hence, educational policies, that are driven by the idea "The lower the cost, the better the education", promote instructivism to the place where it is today. Instructivism is still the lowest in cost [5], as it was centuries ago.

Besides the cheapness, there are other reasons to apply instructivism in educational settings. It is also effective. If more that half of the world's population knows how to read and write, without any question, it would be a great achievement. The "Project Follow Through", one of the largest educational case studies, also shows this [5]. "Project Follow Through" is conducted on more than 79000 K-12 students and examined the progression of them for seven years. It measures the effectiveness of different pedagogic issues. The students are compared with respect to their ability in reading, arithmetic, logic, etc. as well as higher-order thinking, and self esteem. The results show that instructional education surpasses the other theories in most cases. The only exception is student-centered (constructivist) test subjects, which scores significantly better at testing the ability of integrating ideas.

Although the idea of instructivism is fading away from the field of theory slowly, it seems that the instruction-based education will be dominant at the classrooms for some more time. Hence, instructivist perspective also affects the computer-aided education. One particular field of studies in computer science concerning the instructivist views is the Intelligent Tutoring Systems (ITS). ITS's will be covered in chapter 3 in detail.

2.3.2 Constructivism

Another major educational theory that needs to be discussed is constructivism. Constructivism is not a new concept. It is a philosophy that has been applied to sociology as well as cognitive psychology and pedagogy. Even before Renaissance, the ideas of constructivism can be seen in literature. Upon arguing the dogmatic influence of the Catholic Church, philosophers created the foundations of constructivism. These foundations are based on an idea Giambatista Vico commented in 1710:

"One only knows something if one can explain it".

This idea became the starting point of constructivist theory. Immanuel Kant improved this idea and concluded that human beings are not passive recipients of knowledge. Learners actively take knowledge and connect it with their previous knowing to make it a part of their cognition. [6] After 1930's, these ideas found their place in the field of cognitive psychology.

Most of today's constructivist theories are based on the work of the Swiss scientist Jean Piaget. He worked on human cognition and child development. He divides cognitive development into four major stages: sensory-motor, pre-operational, concrete operational, and formal operational stages [7]. The child understands its limits and the physical reality, creates his/her own perspectives, reorganizes these perceptions, and learns to make inferences and predictions in these stages respectively [1]. The following ideas lead to modern constructivism:

1. Knowledge is constructed, not transmitted.
2. Prior knowledge impacts learning process.
3. Initial understanding is local, and cannot be modified by external factors.
4. Creating meaning requires activity. [8]

Focusing on a pedagogic view, in constructivist perspective, meaning is tightly related with experience. Each student has their own set of experiences and knowledge based on a distinct past and world-view. These past experiences can be correct or incorrect and even absurd. With new information and experiences, the student challenges his/her old knowledge, and re-constructs past knowing. In doing this, he/she relates new information with past experiences in memory. The more related subjects remain, while less related subjects are vulnerable and forgotten quickly [6].

Constructivists believe that knowledge is constructed, not transmitted. Each individual has a separate world-view. This view consists of the individual's representations and is constructed by his/her past experiences. Knowledge construction is a natural process. Whenever someone faces any kind of matter (This can be a problem, a difficulty, or anything the person does not know or is not aware of.) he/she attempts to explain it with what has already been known.

The constructive perspective opposes instructivism at this point. According to the constructive perspective, since knowledge construction is a natural process, a teacher cannot directly transmit knowledge to a student. The teacher cannot know how the students perceive the subjects whereas the students cannot think in the way the teacher perceives it. This is because all individuals have different experiences, thus knowledge, which is not shared. Even if they share the experience within the classroom, the interpretations of subjects will be different. And since the interpretations of the subjects are different, the teacher can only make the students memorize the instructions proposed by him. Therefore, the role of the teacher in a constructivist classroom should be creating a workspace where each student constructs knowledge individually. In doing this, the teacher must guide students with motivating subjects and become source of knowledge for individuals.

Knowledge construction results from activity, so knowledge is embedded in activity. The knowledge and the contexts cannot be separated. One cannot respond only to information. The response is generated when one experiences the information. One can memorize that two times two is equal to four, as we did in early stages of primary education. This becomes meaningful only when the student experiences two boxes each containing two seeds. Learning and memorization are different aspects. One means integration of information to real life by the aid of experiences, while the other is just holding information somewhere inside.

Knowledge is anchored by the context. The knowledge that we construct includes information about the context of experience. If we had an embarrassing experience during studying, this experience becomes a part of the knowledge. Eventually, it is possible that one uses this experience to recall the actual content. The constructed knowledge not only consists of information (content) but also the facts about the time and consequences in which it is acquired [9].

Meaning is in the mind of the knower; therefore there are multiple perspectives on the world. Each individual has a unique process of knowledge construction, integration and perception. This is because each individual has a unique set of experiences, which in turn creates a unique view of the world. This does not necessarily mean that two people cannot share world-views. Any two can share their views. This process of sharing world-views depends on challenging or negotiating two different perspectives. One can also construct a meaning from another's world-view and combine this with his/her own. The resulting construct is not the copy of the gathered meaning, but an integration of two meanings in a way that also differs from person to person.

Knowledge construction is based on a problem, question, confusion or a desire for knowing. Because of this, it involves personal ownership of that problem. What produces the knowledge construction process is the relation between what is known and what is observed. Man tries to find alternative explanations to questions that are not answered by the past knowledge and experiences. Meaningful learning starts with a question and a search for an answer. The learner must create this question individually. Learners should actively seek answers to questions, which in turn, create an ownership of the problem. In other words, knowledge that is learned as an answer becomes a part of the learner. The ownership of knowledge makes the construct more relevant, and important to the learner.

Knowledge construction requires representation of what is learned. The activity and experience is the prime requirement of meaningful learning. While alone, they are not sufficient. Learners should share and represent their knowledge in order to repeat the knowledge construction process, criticize and learn if the construct is solid. Students not only generate more realistic and meaningful knowledge but also any wrong knowledge will be used to create more solid and relevant constructs by revising what they have done [9].

Most of the results show that teacher-centered education and student-centered education gives close results in all aspects, although it is expected by the constructivist theory that the results should be significantly better in student-centered education models. (Although results show that student-centered approach perform slightly better). The constructivists claim that the teachers mediating the constructivist classroom should be constructivist at first. They propose that with a well-established constructivist view, the student-centered approach can do significantly better later. As it is evident, being a teacher in a constructivist classroom is a hard task. The teacher should be a mediator in order to encourage self-learning, and protect each learner's right to reflect his/her knowledge equally among others; an expert in order to support information as the students construct knowledge; a philosopher in order to raise questions and a desire to seek the answer in students' hearts; and a learner in order to guide, not to instruct learners in the process. This is why constructivist theories have failed to make an impact in modern education systems while dominating the educational theory. The constructivism requires more intellectual experts than instructivism does. However, it is also clear that today's education is not appropriate for real life. Critical thinking and knowledge integration is neglected by instructivist ideas. As a result of this, fearing from lack of capacity for real life becomes a psychological problem among graduates. It is possible that, few decades later, with constructivism in practice, more intellectual experts can be raised and thus education becomes more appropriate for real life.

As constructivism views the learning process as an individual activity, it lacks explaining the social effects in learning. World is a social environment and education should built emphasis on this. Jonassen, Peck and Wilson [9] points out that the students in even constructivist classrooms tend to agree upon shared ideas, which are discussed and created by a group of students. This means that social relations are also changing the way learners think *Social constructivism*, a branch of constructivist perspective, tries to explain the knowledge construction process not as an individual but rather a collaborative activity where individuals are involved.

2.3.3 Social Constructivism

While Piaget emphasizes that the primary role in education is individual actors living in a social community, this definition experiences difficulties in explaining the social effects in education. In contrast, Vygotsky and socio-cultural theorists presented a perspective that strongly emphasizes the social effects. This view perceives the social interactions between students that actively create knowledge as a primary effect in education, while individual construction of what is shared and gathered is of secondary import. According to Vygotsky, pupils tend to regulate each other's attitudes within a community. When working on a complex problem, pupils assist each other just as adults do. By assisting and working collectively they can solve difficult problems that they cannot solve individually [10].

The greatest distinction between Pigaet and Vygotsky's work is in explaining the cognitive development of the child. While Pigaet emphasizes that in early years of life, the child explores its' boundaries, Vygotsky explains this as a social driven development. He highlights the fact that in early periods, children learn behaviors and their environment by simply copying the activities of their social partners (his parents, other children etc.). In other words, according to social constructivism, the education is a set of social experiences, exchanged and constructed between individual social actors [7]. While Pigaet's theory places individuals as a victim of social relations, the social constructivist theory places the individual as a center in social interaction.

In terms of education, social constructivism favors collaboration between learners. Learners should actively exchange knowledge in order to create meaningful and usable knowledge. The knowledge making process is still a part of the individual actors. As learners interact within knowledge building communities, their knowledge is influenced by that community [11]. The knowledge of that community becomes a part of the learner while this knowledge is affected by each of communities' members. The communities of learners can be seen as a distributed memory, where each learner stores some of the collective information. This distributed memory has more dynamism and capacity than any one learner has.

As Voltaire stated, today, even an eight-year child knows more than Aristotle [12]. Problems that we face in everyday life are bigger than only one person can cope with. The education system needs to somehow embed group working and collaboration concepts to the learners. Collective learning strategies must be encouraged.

2.4 State of the Current Education System

Although nations spend more and more money on education, researches show that the average success rate is constantly dropping [4]. More than that, everyday more learners are added to the educational system. We are living in a knowledge age, and the level of knowledge is rapidly increasing. In order to cope with this situation, more and more experts, classrooms, money and time are required. Today, students should get an education required for catching up with the world's level of technology and progress, and as a result each day, the curriculums of lectures get heavier and heavier. Students graduate from schools filled with irrelevant information, much of which is unusable in everyday life. This creates adaptation problems in the after-graduation period. In contrast, little has changed in the organization of the educational system. The education is still based on the instructivist idea that contents are divided into lessons and directly given to learners.

One major limitation of the current educational system is the evaluation tools employed. These evaluation tools are weak, inappropriate and outdated. The questionnaires, multiple-choice and true-false tests used for evaluating the attitudes and knowledge of the learners, and most of these test results have near zero correlation with real life performance [1]. Most children lack motivation because of the evaluation methodologies. They observe that memorizing context for a short term is sufficient for being scored as a good student. Such an attitude is unacceptable for education and the blame is on the system. More than that, it is nearly impossible for a trainer to evaluate the learners correctly. At best cases, the trainer only has access to the final product of knowledge construction process. There is no way to trace the development, thus guiding and evaluating pupils with such poor evaluative tools. The evaluation tools should provide new methodologies for determining collaborative skills, constructed knowledge and critical thinking.

The educational system also suffers from the rapid changes in technology. The world changes so fast that graduates may have only outdated knowledge after a few years. No possible instruction organization can get a grasp of such speedy advancement.

Thus, a more flexible education is needed. The system should be able to adopt changes rapidly. This is where constructivist theory goes one step forward from today's education. Since in constructivist learning environments, the learning depend solely on self-learning and individual progression, with the aid of experts, the learners can be guided to learn updated information more dynamically.

As noted earlier, the world today needs individuals with cooperative and critical-thinking abilities. It is doubtful that a teacher-centered approach can cope with the situation all by itself. Evidently, the days where the current curriculums are totally full and desperate to serve the needs of the emerging technological progress is very near. In such a situation the role of the education is not to teach everyone all that is needed to be taught, but to teach how to learn what is needed. In that aspect theoretical studies show that the only way is via constructive, student-centered approaches.

However, student-centered approach alone is also insufficient. There are more things to learn but less time. A careful guidance encapsulated within an authority is also needed. While constructivist point of educational theory emphasizes that the meaningful learning can only be maintained by using individual skills and experience, there is a need for guidance and evaluation. There should have some experts in such an environment, that both evaluate the trainee's performance and intellectual skills and, whenever appropriate, guide them [1]. In addition to this [13] points out that in collaborative learning environments, where students communicate with each other in the whole process of knowledge construction, some students tend to dominate the ideas of others, which is a phenomenon called "Social dominance distractions". Also, some students tend to lose concentration and motivation without evaluation because of poor management strategies between students during the work constructed [13]. Some may take more responsibility than others, which clearly create gaps between students. Instructivist perspective used more than a couple of centuries proved to be successful in this respect. The teacher being at the center of the educational community may guide the learners more effectively to the targets of education.

In short, an ideal educational system should be constructive with respect to individuals, where each learner should know how to create knowledge and build new ideas on the existing blocks of information. Teamwork and collaboration over ideas must be encouraged at all costs as well as critical-thinking. But it must also be instructional with respect to the community. Trainers should guide and drive learners to the source of meaningful knowledge.

2.5 The Theory Behind

Our work in this thesis is to identify an education system and combine technological advances for aiding it. Both instructivist and constructivist theories have strong and weak sides. As knowledge building happens in one's mind, guides and instructors are needed in order to find the true source of knowledge.

The tools of technology, especially computers should be used in order to facilitate education. We believe that a successful educational tool should have the following characteristics:

- A successful tool should enable users construct their ideas. In this construction process, the learner should be able to use any source of external knowledge as well as integrating his/her past knowledge, and representing his/her ideas clearly.
- An educational tool must provide content as well as concepts. Learners should be able to relate learned concepts with everyday life experiences. For meaningful learning, concepts and the contents should be perceived together. An educational tool should provide facilities to enable content-concept relationships.
- Collaboration between learners should be enabled. The concept of group working can only be proposed to learners by experiencing it. However, authority within a collaborative community is also required. As interpretations of each learner differ, the reaction time also differs. The community may

suffer from domination of individuals, not from correct knowledge. Because of this reason, authority is needed in order to protect each learners' right of sharing ideas within cooperative community equally.

- The knowledge learned should be in the first place, not short-term memorization. The evaluation methodology of today's education fails to promote the learning process over memorization. A successful education should evaluate and reward the learners with their knowledge constructs and the process of meaning making.

Chapter 3

Education and Technology

There has been a considerable amount of research in computer aided education related to both constructivist and the instructivist perspectives. Although today, most work takes a constructivist point of view, it is worth covering the generation of instructivist and constructivist educational software together. This chapter is divided into three parts. In the first section, the relations between computer science and education will be explained briefly. The second section explores the instructivist affects and generation of teacher-centered software, intelligent tutoring systems as well as the first ideas of constructivism in computer aided education. In the last chapter, a shift from the constructivist perspective, to social constructivism is presented. A new wave of software; virtual learning environments and in turn, collaborative learning environments (also referred as constructive learning environments) are covered in this section.

3.1 Role of Technology in Education

After the early 50's computers become a symbol of the new technology. Manufacturers continue to market ever cheaper and more powerful of computers. Eventually, most people use computers in their everyday lives today. Schools also gain access to this new technology. At first, computers were considered as a new knowledge that should be given to the pupils. Courses to teach using computers were added to school curriculums. In the early 90's, it became a shame not to have some sort of computer access in every school. In 1996, more than 50% of all schools in America not only have computer access, but also have some sort of Internet access [14]. As computers became a part of the educational system, theorists started to think using the computer within the educational system. Feldhusen and Szabo refer to this new technology as an "educational heart transplant" [15]. The idea of computer-aided education became so popular that some theorists started to perceive computers as a medium for education. As Bork [16] commented:

> "We are at the outset of a major revolution in education, a revolution unparalleled since the invention of the printing press.... By the year 2000, the major way of learning at all levels, and in almost all subject areas, will be through the interactive use of computers. (p. 53)"

Although this quotation is a little exaggerated, it emphasizes the importance of computers in education. Ever since the computer and education concepts become related, aiding education with computers becomes an important part of computer-mediated studies. As Internet access becomes widespread, a new form of computer-mediated learning appears: collaborative learning. Collaborative learning aims to generate workspaces for a more affective education. Pedagogic struggles also gain a new medium of argument: the way computers support education. The educational tools are divided with respect to their educational perspectives, and also categorized as instructivist and constructivist. Although most of the work in computer literature claims to have a constructivist view, this is mostly because of implementing a software with a teacher-like authority is not a simple task. As we shall see, the first implementations aim to have this property.

3.2 Computer as Tutor

It was 1950's when the computers meet with education. The teaching machine that directly instructs became a popular idea. This technology was not something that teachers only illustrate their teaching; it was thought that this machine could take the places of the trainers in the classroom. In 70's when the first mass-produced microcomputers started to appear in the market, both researchers and the press began to talk about the computer-as-tutor metaphor [7].

The educational software in these days mostly depended on a behaviorist view. The theory behind them relied on the teacher student dialogues. Most of the educational software and classroom interaction can be characterized as I-R-E sequences. That is initiation, followed by response and finally evaluation. The process happens as:

Teacher: If it is a pentagon, how many sides does it have?

Pupil: Five.

Teacher: That's right.

Human to human interactions as in this dialogue are copied into software and the "Computer Aided Instruction" phenomenon appeared in educational platforms. Computer has the role of the teacher as pupils interact with the software directly.

The more advanced versions of computer aided instruction tools are called "Intelligent Tutoring Systems" or ITS. The goal in this approach is strongly associated with the computer-based education: creating a fully individualized education [17]. The intelligence of an ITS system in that respect is not its knowledge expertise. In order to create a computer-driven classroom, in addition to knowledge, the system should create an intelligent tutorial dialogue. If that is the case, it was possible that computers took the trainer's places in classrooms years ago. However, the system should also diagnose the learners' needs and generate the tutorial with the light of this intelligent evaluation. ITS fail in that respect, because it is difficult to construct and maintain a system that effectively diagnose student errors and needs. The ITS's are mainly used in military and industrial training concepts. This shows how complex and costly such systems can be. It is clear that financial requirements of ITS's are beyond the reach of schools.

As intelligent tutoring systems' complexity and high cost became more evident, the computers in the place of tutors idea became less referred. ITS lost its popularity until another invention in computer-mediated platforms appears: The Internet and its widespread use. Tutoring systems again came to surface. Since this is not in the scope of our work, these exercises will not be mentioned here. As the constructivist view takes place of the behaviorist view, an apparent shift from teacher-centered to student-centered practices take place.

3.3 Virtual Reality and Education

One revolutionary movement in computer-aided education can be seen as the introduction of virtual reality. In "Mindstorms" (1980), Papert denies the traditional pupil under control of the computers idea, while favoring an education where computers are controlled by pupils. He states that children might well be more engaged by information managing activities if what they discover describes something immediate to their own experience. According to this idea, the pupils might do this by gathering learning experiences from the computers. He introduced this idea with microworlds. In microworlds, the children can apply their knowledge and get the results as a creative activity.

Papert shows the results of this idea with his turtle, LOGO. LOGO is a vehicle for introducing certain concepts of writing computer programs. Within LOGO, user can write a program that causes the turtle to act according to the instructions supported by the user. Unfortunately, LOGO requires more that simple programming. Although LOGO is not very successful in that aspect, as a first representation of microworlds, LOGO became a brilliant success story.

Indeed, microworlds and virtual education (VE) idea became very successful. The students can not only see, but also by the aid of these virtual worlds, actively engage in activities that are too complex or impossible to see in the real world. Dede, Salzman and Loftin [18] commented this as "...By becoming a part of the phenomenon learners gain direct intuitions about how real world operates."

The idea of microworlds is applied to certain aspects of education after its introduction. ScienceSpace is one of the most referred groups of microworlds designed to aid students in challenging concepts of science [18]. ScienceSpace consists of three microworlds: NewtonWorld, MaxwellWorld and PaulingWorld. NewtonWorld is an environment for investigating one-dimensional motion. MaxwellWorld introduces the concept of electrostatics, while PaulingWorld studies the molecular structures and their representations. Another notable work is "Project: A.I. Wars". As an ongoing project for more than ten years, the intention is more or less the same with LOGO. To teach children aspects of both structured and object oriented programming [19].

Another aspect of computer-mediated work in education is the Virtual Learning Environments (VLE). The idea of Virtual Learning Environments emerges with the computer-oriented communication tools. VLE are on-line domains that allow collaborative interaction among teachers and students as well as providing learning resources to students at any time. The collaborative interaction among users provided is by computer communication tools like e-mail, bulletin boards, newsgroups, teleconferencing etc. [20]. Most of today's VLE are based on client server architecture with private access protected by login identification and passwords.

The objective of virtual learning environments is not to produce a copy of a classroom on-line, but to provide tools for learners to facilitate their learning process. Another aim is to create an environment where users learn collaboration and sharing of knowledge.

A VLE introduces many educational properties at a time. Once a user is logged in, he/she can view course materials created by the instructor, work on cooperate projects within groups, communicate with group discussion in a virtual lobby, access referenced materials and involve in private conversations with the instructor or other users. Obviously, not all VLE's do support such an extensive list of features, though a combination of these elements exists in most [21].

Most of today's VLE user interaction is done with a text-based interface. Despite this text-based nature of VLE's, the environment is divided into components. These "rooms" are connected to each other logically with virtual "paths". Users can walk within these virtual paths to enter communication rooms, go to information libraries, or even to talk with the instructor.

However, VLE's are more than a communication environment. Each virtual space in a VLE may also contain programmable or changeable entities. These objects can be questions, guides, lessons, handouts etc. These entities can also be programmable in some VLE's that they can respond to user interactions. Whether the VLE is Web based or not, the primary form of them is MOO, (MUD object-oriented) or MUD, (multi-user domain). While MUD is a text based virtual reality in which users can interact and access to resources, MOO also enables its users to actively build and extend the virtual environment. Ability to create new rooms and objects, and shaping them with individual preferences makes a MOO an incredibly powerful and flexible educational device.

The NICE (Narrative based, Immersive, Constructivist/ Collaborative Environments) project is one of the first multi-user learning environments reported in the literature [22]. The project aims at K-12 children and based on constructivist theories. In Nice, users, guided by a narrative, collaboratively build virtual gardens.

LearningSpace [23], which is a collaborative derivative of ScienceSpace project, is also another notable version of current VLE's. The basic architecture of LearningSpace consists of five main databases for regulating knowledge construction:

- The Scheduler, which provides a schedule manager.
- The Media Center acts as a knowledge store.

- The Course Room, which is a conferencing tool that allows public discussions and group work between learners.
- The Profile Manager contains basic information about both learners and tutors similar to a homepage or an on-line CV.
- The Assessment Manager, which is an instruction tool for private testing, and for giving feedback on participants' work. It also allows a grade book view of all participants.

LearningSpace includes tools for browsing the web and inserting multimedia material into LearningSpace documents. Asynchronous communication tools in LearningSpace are based on e-mail, which is used for private one-to-one discussions and the Course Room where threaded private or public discussions can be done under the authority of an instructor. Synchronous communications are supported by a variety of tools supported by "Learning Server 2.0" such as chat, whiteboard, video and teleconferencing. In addition to these, resources and other content may be exchanged via the Media Center. Students can maintain a process folder, which can be seen as a private room. This folder contains record of learner's work and assignments throughout the course. Completed courses may be archived by the instructor for future use. A portfolio is contained in every user's profile in Profile Manager. This is a secure area for returned assignments, which can only be viewed by the user that posted the assignment and the tutor [lotus].

Other than LearningSpace and NICE, there are many other notable VLE's. Some of these are WebCT, TopClass, Virtual-U, ARIADNE and CoMentor [20]. These works will not be presented here as they all consist of more or less the same elements as LearningSpace.

3.4 Discussions

As an organizational tool for knowledge construction and organization, the work reported in this thesis should support constructivist theories. Learners, at the center of the knowledge construction process, should build new ideas and relate them with the old experiences. The trainers should guide learners to the knowledge source as well as being a source of information. Authority should be used in order to regulate cooperative work and drive learners to the required knowledge, not to enforce material.

Collaboration is another aspect of constructivist frameworks. Collaboration within the groups should also be supported. The practices on VLE show that in a computer-mediated education, collaboration through content clearly plays an integral role in learning. VLE and its proposed communication methods have remarkable effects on cooperative working. The authority implementation, private study facilities and collaboration features of VLE can be a basis for an educational tool. As VLE examples demonstrate, administration abilities and support for both individual and collaborative facilities of VLE best fit with the theoretical side of this study. As discussions in chapter 2 suggests, our aim is to build a knowledge representation tool where authority and both private and public study facilities are supported. In that sense, the theory of an educational tool presented in this thesis resembles VLE in many aspects. In fact the work here can be seen as a graphical virtual learning environment.

Chapter 4

Concept Mapping

The theoretical side of the work done in this thesis is mostly based on social constructivist ideas. Since learning is an individual process, as social constructivism suggest, cooperation and discussion is important aspects of learning, and should be encouraged. In other words, education setting must support both individual working and cooperative knowledge construction. The discussions in the previous chapters show that a sufficient tutor-based view is also needed, as authority is required in order to shape education on a collaborative platform.

However, theoretical case study is not sufficient alone for building knowledge organization software. User should also have the best practical methods to construct new knowledge. As Buzan commented [24], the traditional note taking, or in other words text based note taking methods are weak and inaccurate for knowledge construction, as text based notes are hard to remember, and time costly to build. Another possibility is using semantic networks, or concept maps as a tool for visualization and production of knowledge.

In this chapter, concept maps are introduced as a basis for our knowledge organization tool. The chapter is divided into four sections. In the first section, the concept maps are described and discussed in detail. The second section is devoted to the relation between the concept maps and knowledge construction process. The third section the evaluation on concept maps is discussed in an educational perspective. Finally, the last section explores the uses of concept maps in computer platforms today and discusses significant examples of concept map use.

4.1 What is Concept Mapping?

Joseph D. Novak developed the concept-mapping idea in the early 70's as a method for organizing information. Concept mapping can be described as a process through which trainees, using past knowledge and perceptions, create a map by using keywords and drawings that represent a specific context. This map includes concepts, which are related with the context.

Concept maps are visual languages that are used to represent and understand thought [25]. The aim of a concept map is to identify the concepts and relationship between these concepts as a structure that directly refers to the drawer's reasoning process. Most of the concept maps use a graph-based diagram. In this graph, nodes (vertices) represent concepts while arcs and links (edges) represent relations. Visual attributes of graph members identify the significance of the concepts and/or relations. More complex concept mapping languages also include structures for improving the user's perception [26]. A group of concepts is an example of these structures and enables the users see both a detailed and a general version of the content at the same time. But basics of a concept map have two visual properties, nodes and links. Figure 4.1 shows a concept map of this thesis' organization.

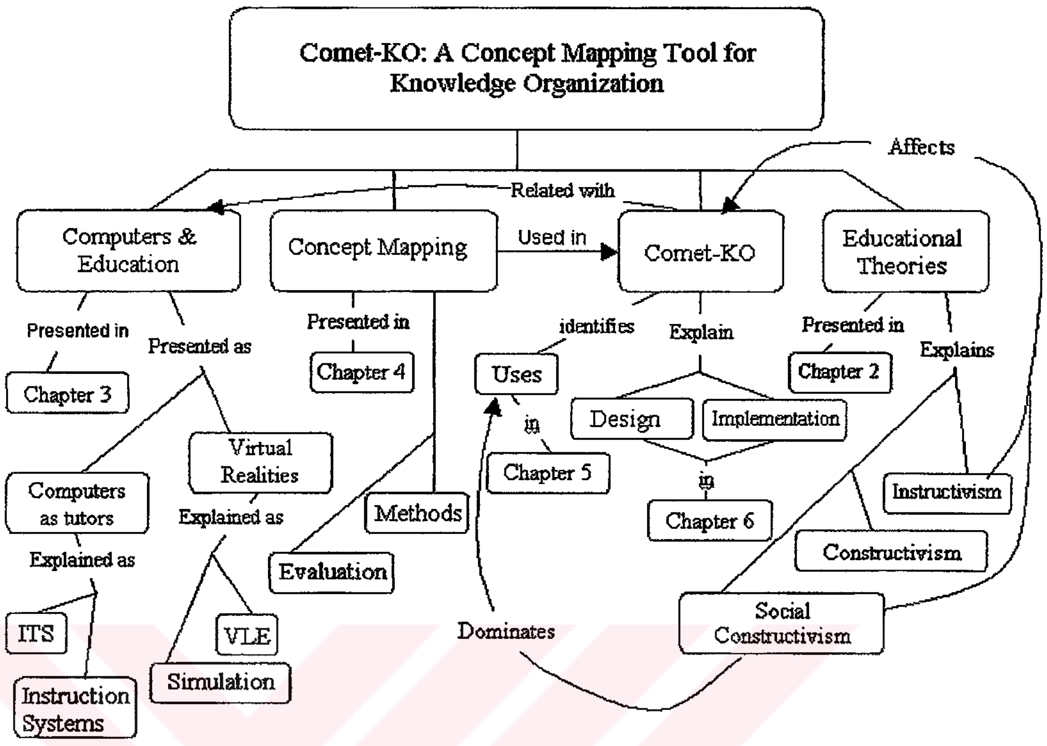


Figure 4.1 – Concept Map of Thesis

Compared to the traditional note-taking model, concept mapping has outstanding properties [27][1]:

- It clearly defines the central idea.
- It clearly indicates the relative importance of each idea.
- It represents the key relations between ideas. This is particularly important for creative work.
- It is space efficient. Many ideas can be presented in one page.
- As a result of the above, and because each map will look different, recalling and reviewing a concept map is easier.

- It is easily extendible. Since it is a graph like diagram, adding a new idea to the concept map means adding a new node representing the idea and its' relations with the already existing ideas.

- Concept mapping enables its creator and reviewers to see the ideas it contains from different viewpoints.

- Concept maps enables contradictions and gaps in the knowledge to be seen more easily, and in this way, concept mapping provides a foundation for questioning, which in turn encourages discovery and creativity.

Clearly, concept maps are excellent tools for building large projects and organizing thought. Since they are easy to draw and visually appealing, it can also be a perfect representational tool for educational purposes.

4.2 Concept Maps as a Tool for Knowledge Construction

Concept maps provide one of the easiest ways to learn content, and is one of the most popular knowledge representation tools. Whenever students are studying and analyzing new content in order to understand and construct upon it, concept maps provide a rich visual tool for identifying the structure of ideas of the context, which is essential for understanding the information. Because they require learners to analyze the underlying structure of ideas they are studying, concept maps are tools for consciously organizing what the learner knows by explicitly describing the concepts and their relationships [9]. Concept mapping, namely semantic networking, specifically engages learners to relate the past knowledge and new ideas, which is the basis for meaningful learning.

Learners who build concept maps actively seek important ideas from information sources. These ideas are needed in order to construct the concept map. Learners also evaluate information for its relevance to the subject, and the way that it fits with other ideas. Thus, concept maps are also constructive visual representations of ideas. Learners create intentional strategies for studying course content by working on concept maps. The concept mapping process requires students to reflect on what they know and to determine how new ideas are related with past knowledge.

Concept mapping as a classroom activity differs from individual knowledge construction by many aspects. Team concept mapping improves individual thinking abilities as well as providing collaborative experience. In one study of knowledge construction, it was observed that many of the learners' maps at mid-program focused on the classrooms' cooperate organization, while at program's end, many of these maps are centered on self-regulated learning [13]. At the very beginning, are constructed by a cooperative view of the whole classroom. The first development stage reflects this collaborative consciousness. Initially, the ideas and relations are created with all learners' contribution, while after this, a "quiet" phase [28], in other words an individual thinking period, dominates the final shape of the maps. This quiet phase involves learners' analyzing and correcting the cooperate-construct with their own consciousness and world-views. Thus, it can be said that collaborative concept mapping empowers both collaborative and individual thinking abilities.

The relation of concept maps and their effects on the student's learning process is the main topic of this discussion. While working with concept maps, students must actively search and evaluate new information. To create a concept map, learners first need to identify the important concepts related with the map. This means reading through texts, lecture notes, and other information sources to identify and evaluate the ideas that are important to understand the content. Rather than defining these concepts, students need to describe how they are related with other ideas and with the whole content. This means that students are not working on separate concepts of content one at a time, but as a whole. Concept mapping provides students a way of relating different aspects of a subject and enable them study all the concepts together. Learners also need to decide how relevant and descriptive these ideas are. In this way, students not only study about the content, but also analyze and criticize, and as a result, understand, and relate it with other ideas.

4.3 Evaluation on Concept Maps

As explained earlier in chapter 2, evaluation process is an important part of the educational settings. Either the information given to the learners or the creativity and critical-thinking abilities are evaluated, evaluation of target's performance is an integral part of the education. The traditional testing methods like true false or multiple-choice exams can only evaluate a small portion of the knowledge in an instruction. Moreover, they do not possess the capability of testing the creativity or effective performance of the learners. Concept maps, by means of educational tools, are also powerful evaluative utilities. Concept mapping activities have proved to be accurate means of representing one's cognitive structure. They can be used to identify learners' cognitive structuring and ability to pattern relationships between concepts [9]. Also, completeness and accuracy of learner's concept maps are tightly related with learning outcomes; like performance, problem solving, and critical thinking [29]. Thus, as an evaluative tool for meaningful learning, concept maps have high-quality capabilities.

Using concept-mapping tools for devising tests also eases the work of evaluators. By simply subtracting some elements of a map, a new test can be devised. As students reconstruct the maps, they embed their reasoning process and relational abilities to these maps. The trainer can evaluate the resulting products and conclude about the trainees' meaningful learning abilities.

4.4 Uses of Concept Maps

Today, concept maps, as a powerful tool for representing knowledge and ideas, are used widely in both educational platforms and corporate settings. The application area of concept mapping varies largely. Several companies use concept mapping for illustrating organizational and management information [1]. Weather forecasters in the Navy that are assigned to the Gulf Coast region use the concept maps to learn about local forecasting [30]. Many universities throughout the world are using the concept mapping to create and organize contents for computer-mediated learning. The software market also emphasizes concept mapping. Several software packages are in the market for professional and individual use today. Mind maps, as a version of concept maps are marketed for improving learning and recall processes.

CMAP Toolkit, an ongoing project in the Institute of Human and Machine Cognition Department of the University of West Florida, is one of the concept mapping tools that is already in professional use. CMAP toolkit allows users to create several concept maps, where nodes of the concept maps can also hold links to other concept maps and multimedia documents. In that sense, a strong grouping mechanism is also generated in CMAP architecture. These concept maps are integrated with the Web and can be accessed by using any browser. The Center for Mars Exploration at NASA Ames Research Center is using the software to organize a huge amount of information on Mars. Also the Storm-LK (Systems to Organize Representations in Meteorology Local Knowledge) project uses concept maps for weather forecasting in the gulf region [30].

KMAP is a classical example of concept mapping tools [31]. KMAP provides a graphical interface for nodes and arcs (concepts and relationships) that can be programmed by the user to create a concept map. Multimedia material can also be added into the maps drawn by KMAP. Although research is currently going on in order to integrate KMAP concept maps to World-Wide-Web by a viewer interface, currently KMAP and all collaborative support are only available for Apple Macintosh.

Another concept map related project is PROLEARN [32], which is an information navigation facility where content is prepared by using concept maps. The PROLEARN software consists of two separate interfaces. The concept-mapping interface can only be used by trainers to construct the course material, while the navigational interface is used by trainees to browse through the content. Each concept is represented as a separate page where other subjects are presented as a list of related concepts. Trainees navigate through the content by selecting the next concept they want to explore. Although it again uses concept mapping in theory, the visualization of content is based on a textual representation, that is, the visualization of the maps is the weak side of PROLEARN.

Mind Manager and Visimap are mind-mapping tools with can be also considered as a sub-group of concept mapping utilities. Mind maps are a special case of concept maps, where the main content reside in the center and supporting ideas branch from the center relative to their importance levels [24]. These branches constitutes to details and associated ideas of the main context. While concept maps can contain more than one main idea, organization of mind maps prevents this property.

Another notable concept mapping tool is Rememex [33]. Rememex is a single user interface version of concept mapping tools with extensive mapping capabilities. Nodes can represent texts, images or RTF formatted files that can be embedded into Rememex maps. Grouping support and collapsible grouping property gives users different views of any constructed idea. In terms of visualization, Rememex proves to be an excellent version of concept mapping tool, with its auxiliary properties. Unfortunately, the project is not fully complete at this point.

There are many other concept mapping tools and projects like JKSI Mapper, Inspiration, Idons-for-Thinking, and Belvedere and so on. The motivation and uses of them are knowledge construction and representation. The representative properties, multimedia support, and collaborative support of these tools vary. As an example, while Idons-for-Thinking and JKSI Mapper tools are single user interfaces, Belvedere and Inspiration have extensive collaboration support. As a conclusion, several concept-mapping tools are already in the market today. Our goal is to create a new concept-mapping tool, with the best properties of these tools as well as generating new ideas for facilitating the user's knowledge construction process.

Chapter 5

About Comet-KO Software

The theoretical basis and the representational features of the program reported in this thesis were discussed in the previous chapters. As a result of these discussions, facilities that must be built on a knowledge organization tool are identified. This chapter is presents the possible uses of our work as a knowledge construction tool, and some important features of the intended architecture.

5.1 The Comet-KO Software

The Comet-KO software is a collaborative data organization tool, which uses concept-mapping techniques as the representation style. First of all, since Comet-KO is a collaborative organization tool, the support for a multi-user interface is unavoidable. Secondly, as an educational software, the ease of use and representational power must be considered. The ease of use relates to how the user uses and interprets with the cognitive map created by the software tool. For example, automatic screen organization features could save user's time as well as the user's effort to draw the concept map. The representational power relates to the range of features supported by the software system.

If Comet-KO supports sufficient organization and representation features for a wide range of possible areas, a high user satisfaction could be achieved. As an example, the user might add a node to the concept map as a representation of an ASCII text file, a word file, sound, movies, and etc. all of which can be directly accessed by the software tool. However, care must be taken since some features, like embedding multimedia directly to Comet-KO can only make the software harder to be used by the user. If user selects with which software the movies or sounds will be launched, this could be a better interactive environment.

5.2 The Uses of Comet-KO

Although Comet-KO software is implemented as a tool for education, this is not the only area that it can be beneficial. Possible scenarios in which the Comet-KO software can be used include:

- **Project Building:** Today, with the pace of the technology, companies need to employ project teams rather than individuals. In such project teams, the workload is divided into smaller pieces among the employees. The real problem arises when documentation data for such projects gets bigger. Each piece of data can belong to various types all of which have different representations (i.e. documents, sounds, images and even video.). These information pieces have relationships or hierarchies between them. Standard data storage systems or note taking methods provide little support for representation of associations, and even if some type of support is present, no preview methods (like putting an image or a video clip as a node to a map) are supported. Comet-KO could be used to enhance understanding and storage of such data while maintaining the relationships between any kinds of these work materials.

  The Workflow Management Systems (WFMS) can be considered as large-scale examples where representational power of Comet-KO provide a good platform for distributing the cooperative project data among the project members.

Another more concrete example could be using Comet-KO for providing support for long-term projects. The Comet-KO software itself could be an example of such projects. There are nearly 60 classes all of which have different methods and structure. Think of what happens if another employee expands this project with another 60 classes later. A person who wants to clearly understand what the software exactly do, how does it perform the tasks, and clarify the inheritance routines of this software require more support documentation than just comments on the code. In such conditions, an organized concept map where classes linked with each other with respect to their hierarchies, and comments for each classes' methods and their uses could provide a very beneficial support. Today, concept-mapping tools are already in use for representing the management details and organization information of corporate settings [1].

- **Teaching:** With respect to education, Comet-KO could be used as a constructivist computer educational tool in classrooms, while with user defined authorization rights on maps, teacher-based scenarios can also benefit from such implementation. The instructors can design contents of lectures as a concept map without enforcing the subjects any material, but pushing students to contribute to what is done so far by other students, and even reshaping the course material according to the students' critical-thinking abilities and creative performance. Teacher also can evaluate the progress of students one by one by examining the contributions of each to the new collaborative environment, the concept mapping community.

The Comet-KO software can be used for teaching many different materials. An example scenario for the usage of Comet-KO in the classrooms can be as a supplementary tool for teachers. Think of the Algorithms courses where, the main purpose of the course is to "embed" students how to explain their method of problem solving to a machine by the aid of a programming language. Generally, from scratch, all concepts are tightly related with each other. For example, no one can teach a student the concept of "arrays" without showing the general concept of "variables". However, the referred course material is so large that generally it must be distributed over several semesters. Though this can be a better method than enforcing students learn a programming language in a month, most students tend to forget some of the critical concepts at the middle of the course period. The Comet-KO software can be used in order to represent the close relationships of subjects covered beforehand. With such a supplementary tool, target audience always have an option to refresh their memories as well as an option to go back and recover the weak parts in the whole content.

The Targets of the above example are the teachers, and the students who are capable of deciding what their weaker points about a course are. What about K-12 and even K-9 students and teachers? These age groups can also benefit from a concept-mapping tool. While concept maps are easy to read and draw, they are also closely related with how we think. Think of a geography teacher. S/he draws a concept map of main products of European countries by giving documentation about their economy and product listings, and also movies about these countries. The students examine this concept map and try to figure out what Industries these countries could have. They supply their teacher with own findings and the teacher evaluates the results. The students can contact their teachers with a range of possibilities. They can fill the gaps in the concept map left blank by the trainer, post their results via mail, and if possible vocally. By this way, students learn not only the course material, but also more importantly, gains self-confidence (because they learn the material themselves!) and methodology of how to start a search or a survey, which they will be using for the rest of their lives.

- **Distance Education:** With the raise of the World Wide Web, education is carried into a new distributed platform where the teachers and students are independent from physical boundaries. Now, students and instructors can reside in different and probably distinct places while being together in the virtual classroom. The real disadvantage of such a classroom is that the teacher has little chance to intervene with the students directly. This means that the clarity of the content plays a critical role in the learning process, since it is hard for the teacher to deal with misunderstandings or difficulties of students to understand the content. The

Comet-KO software could be used to enhance the lecture material by aiding students with a more understandable organization of the course material. Since the concept-mapping techniques are closer to the way we think than the standard written notes are, it is guaranteed that the students feel more familiar to the content. Also the teachers may have a way to guide distant students to contribute to the virtual classroom and get their attention to the lectures on this virtual environment where the instructor has no possible tool to measure the level of attention.

- **As an Organization Tool:** More than computer aided education and project management, the Comet-KO software could be used by non-professionals in order to organize their material. The classic supports for visualizing the data on a computer consists of file explorers with most of time messy directory listings without any support for creating relationships between two data objects. If the user has two separate directories one for related images and another for related documents published by one artist, he/she has no tool for relating the contents of these folders other than his/her memory. Think of situations where videos, sounds etc. have a place, and think of situations where there are hundreds of such artists. The Comet-KO software can be used to organize such related objects. If the documents covered are not things like images and movies but more active data updated frequently, the significance of the Comet-KO software can be seen more clearly. For instance, documents are active data that user may want to relate multiple of them, see together and even cut and copy some material from one document to another.

- **Concept Maps:** Concept mapping, or in other words semantic networking, or specifically mind mapping are becoming more popular and marketable as time goes by. Tools to integrate concept maps with computer technology are already being sold in markets. Comet-KO software has all the possible capabilities of these concept-mapping tools.

- **Teaching Use of Computers to Children:** As the Internet grows, a need for teaching Internet and computers to the new generations has become a necessity. The age limits for starting using computers is continuously dropping down. However, nearly no new software targeting children is marketed. The main objective of a software that targets the children as audience must have an environment that is easy to use. It also should facilitate clear understanding of concepts and facilities of the system without having the children messed up with the complex networking rules and properties like sockets, TCP\IP protocols, and databases. Looking from such a perspective, Comet-KO achieve to fill in most of these requirements. More than its role for computer aided education tool, it can be used to make children familiar with the virtual environment as well as being a good starters tool for learning about computers. Actually, it is a far better alternative for this purposes than ICQ or Web browsers, for these tools are the widest used tools among children with no possible contribution to their learning process.

5.3 Capabilities of Comet-KO Software

At first sight, the notable components of the Comet-KO software are threefold. The nodes, the links between two nodes and the grouping facility are the major components. The concept maps are built on a window-based environment. The main advantages of the Comet-KO over traditional organizational software can be summarized as follows:

- **The window based environment:** The main workshop environment fully supports the window-based principles. User can open multiple windows and manipulate them at the same time.

- **Sufficient component types:** Although all can be considered as nodes, several types of components are supported. User can embed images, text documents, and etc directly to the concept map.

- **Arrows and links:** The Comet-KO software supports user for both drawing links and arrows between two components. Arrows can be used to represent hierarchies while links can be used to represent relationships, which is sufficient for all kinds of representations.

- **Grouping:** User can create groups of components for representing a bunch of related data items in a project.

- **Object Oriented Design.**

- **Undo/ Redo Option:** The Comet-KO software also supports multiple actions to be held within the development workspace. These actions can be rolled back with the powerful undo/redo option.

- **Miniature View Option:** The miniature view option is another powerful interface tool by which the user can search through the workspace on a preview-like screen.

- **Launching External Application:** By the aid of this option, the user can embed new data types as nodes into the map and have the ability to access the information within these nodes.

Chapter 6

Design and Implementation

In this chapter, the implementation and the design of Comet-KO is presented in detail. Information storage techniques and capabilities of an ideal concept-mapping tool are presented. The features supported by this tool will also be considered.

Comet-KO is implemented in Borland C++ Builder 4.0 [34]. This selection is done because of the extensive database and visual components support of Borland C++ Builder as well as having the performance benefits of a C++ compiler.

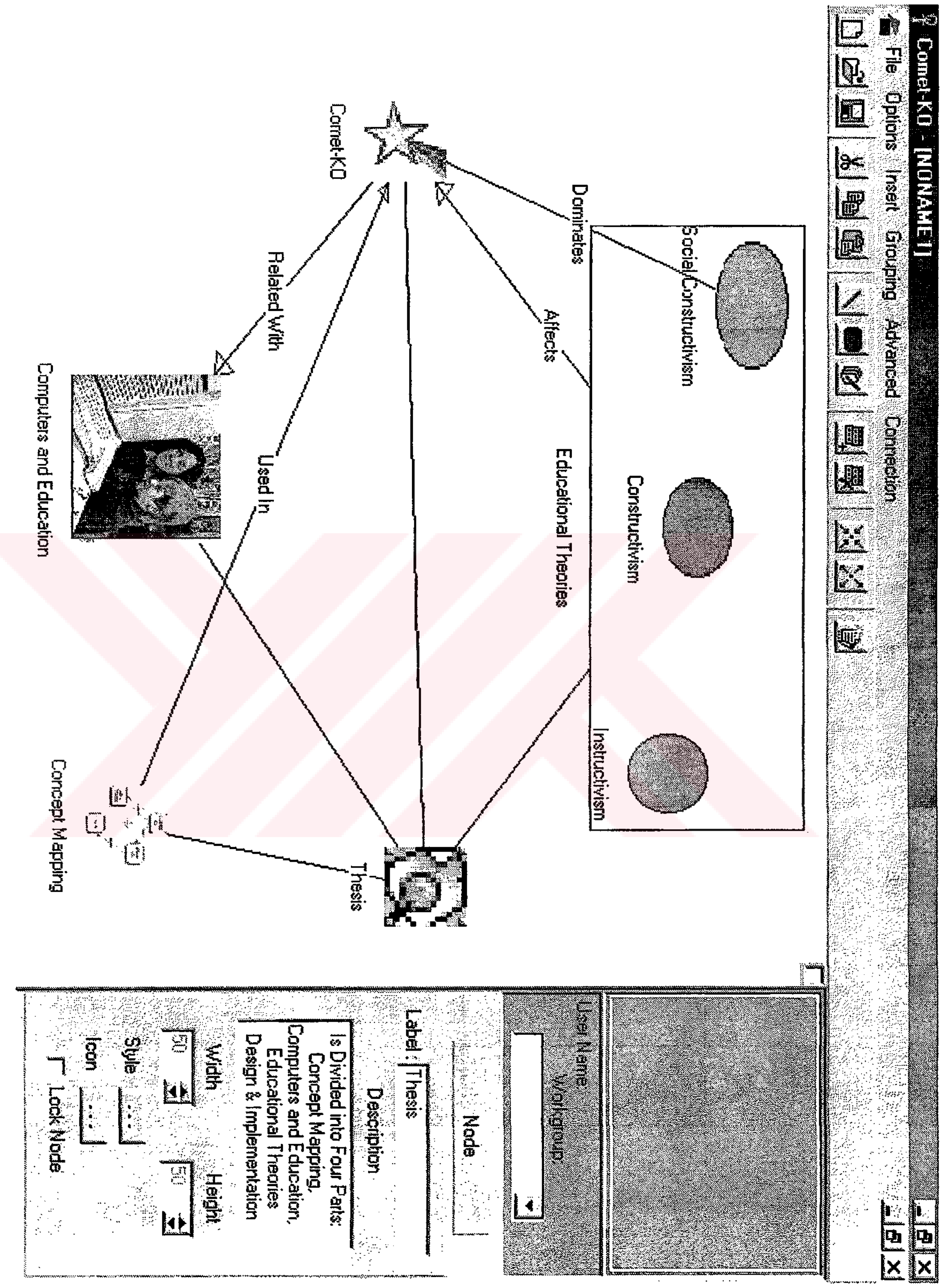
6.1 The General Framework

The aim of Comet-KO is to create a knowledge management and construction environment not only for educational but also for professional purposes. The users should be able to collaborate on this task. This makes a distributed, multi-user environment one of the essential features, however the architecture should be built on the emphasis that standalone usage is also supported. A multi-tier architecture with unicast network topology [35] is used in this work to handle collaborative requirements. According to this architecture, each client is also equipped with a database, which can act as a server in a workgroup session. Clients can both access to a local or a remote database any time within this session.

In the database, a concept map may have different views for each user. Figure 6.1 and 6.2 illustrates two concept map views on the same data. Note that each view represent the same information, while the only difference between these two maps is their presentation. These views are associated with the user identifications of clients. The concept map relates the members (nodes, edges/associations, groups, etc.) of it with different views in the database level. Figure 6.3 illustrates the relationship between two clients and the information residing within the database.

6.2 Features

The figures above describe the interface, database, and communication methods in a session briefly. In order to create concept maps, evaluate the content of these maps, identify and edit users and workgroups, and regulate communication, some tools are presented for the users. These tools have distinct properties and uses. Every workgroup may consist of many clients, one of which also serves as a data store both to itself and to other clients for each session in the workgroup. In that respect Comet-KO architecture can be split into two parts with respect to the capabilities presented to a user: client agent and server agent. In this section, the complete capabilities of Comet-KO are presented, though, at this stage, only some of the presented features of Comet-KO are implemented. Implementation of these features will be discussed in detail, later in this chapter.



Label : Thesis

Description

Is Divided into Four Parts:
Concept Mapping,
Computers and Education,
Educational Theories
Design & Implementation

Width 50

Height 50

Style ...

Icon ...

Lock Node

Figure 6.1 – Concept map of thesis drawn by Comet-KO. Thesis icon selected

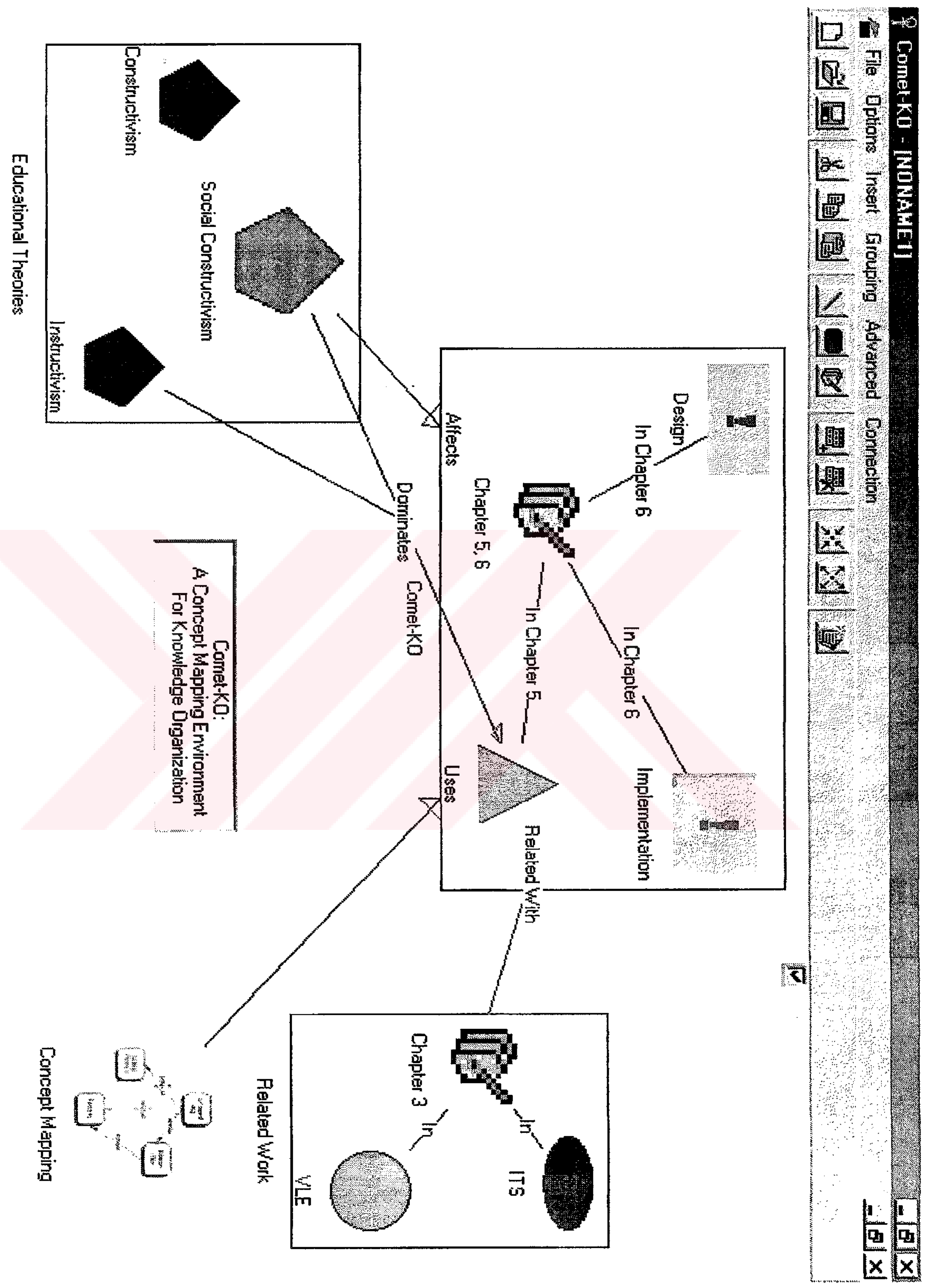


Figure 6.2 – Concept Map of thesis drawn by Comet-KO

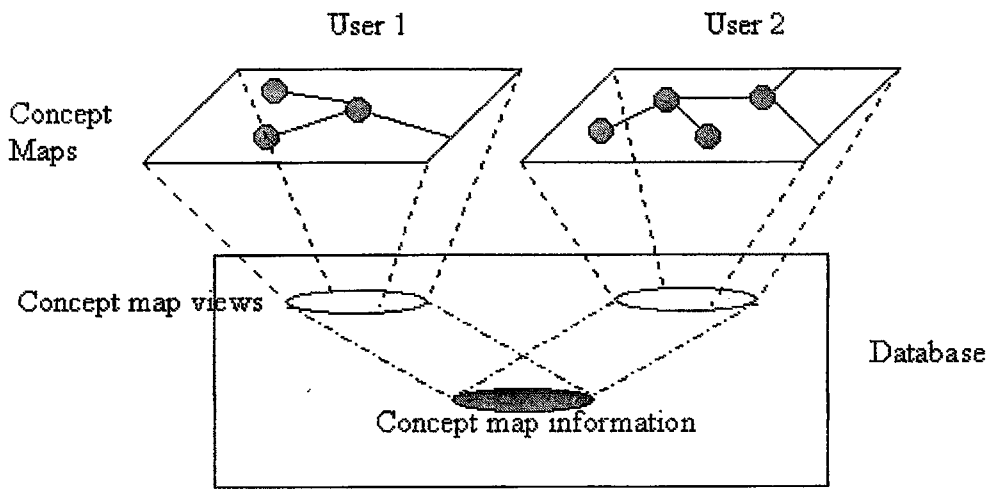


Figure 6.3 – Two users viewing the same information with different representations.

The client agent is the user interface, and other tools that a user employs while interacting with Comet-KO. These tools aid a user to draw a concept map, present maps with different views, organize maps, and query the server data. The client agent is designed using the open architecture shown in figure 6.4.

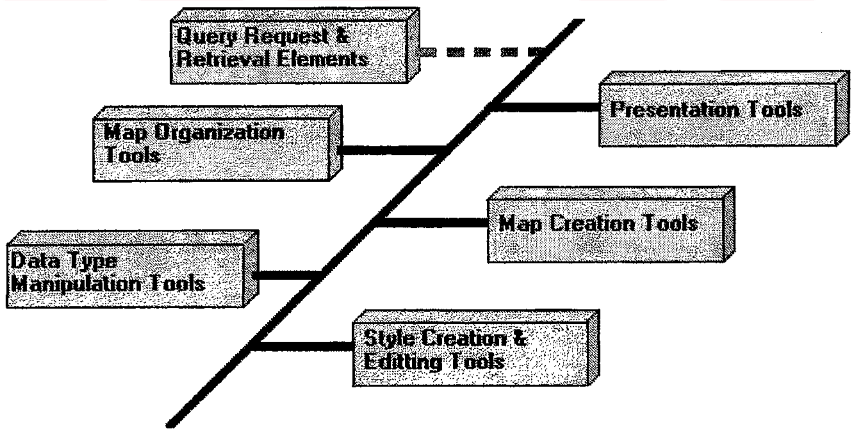

Query Request & Retrieval Elements
Presentation Tools
Map Organization Tools
Map Creation Tools
Data Type Manipulation Tools
Style Creation & Editting Tools

Figure 6.4 – The Architecture of the Client Agent

- **Presentation Tools**: These tools enable the user to view different concept maps, the data of which can be either local or remote.
- **Map Creation Tools**: With the aid of the Map Creation Tools, the user is able to draw a concept map from scratch and add nodes, links, groups and even data derived from other applications. These tools also include options that facilitate map drawing such as cut, copy and paste.
- **Style Creation and Editing Tools**: One of the most powerful abilities of Comet-KO is that the user can create his/her own map elements and save them for future uses. The styles will be discussed in detail later in section 6.3.2.3.
- **Data Type Manipulation Tools**: In Comet-KO, some data types like images and documents can be used directly as nodes in a concept map. For other data types, like C code or movie files, this ability is not supported. However, if the user wants, he/she can add such data types in order to allow Comet-KO recognize and accept it as a node type. Such facilities are provided by the Data Type Manipulation Tools.
- **Map Organization Tools**: These tools automatically layout the map for optimum viewing.
- **Query Request and Retrieval Elements**: Since different clients are connected and share their information with a server, the server holds precious data, which must be evaluated. Though all clients do not need to have such ability, some of them should. By the aid if these tools, the user may request comparisons, and filtering on maps from server-side.

Not all the component tools will be needed by all users. For example, the query request and retrieval tools are not needed by a student user who only participates in the building of a concept map, though they will be essential to another user who evaluates different concept maps, and finds similarities and differences between them. These observations led us identify different client agents for different purposes. There are three essential and obvious agent types with the following capabilities. One client agent has access to only presentation tools. Such agents may be integrated with in a Web browser and can only view, not create or change concept maps. A second agent type has more capabilities allowing the user to view and draw concept maps. Presentation tools, map organization tools, data type manipulation tools, style creation and editing tools and map creation tools are components of this agent. However, it lacks query request and retrieval elements. This component is left to the third and the complete form of Comet-KO agent. Each agent type has increasing capabilities and higher complexity. The first kind of agent can be used for presentation only purposes, while learners, as collaborators use the second one. Only users deeply involved with a concept map creation process uses the last kind of agent.

The server agent is the interface by which the client agent interacts with the database. The server agent is responsible of maintaining the global database, running the queries submitted by the clients, and handling the client connections. Figure 6.5 summarizes the server side and its elements.

- **Global Data Warehouse**: Is the database that holds information of concept maps.
- **Data and Client Identifications**: Each session must somehow hold information about the users and their concept maps, and can identify them and their sessions. This option is presented in order to enable identification of users as well as adding removing or editing these user identifications and their authorized rights on the workgroup.
- **Querying and Information Retrieval Tools**: The server-side information retrieval systems and querying tools, which handle client-side requests.

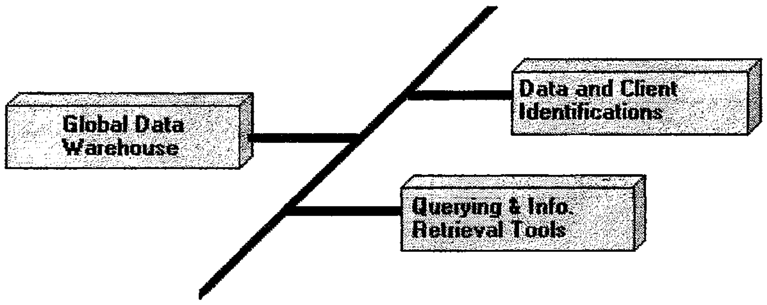

Global Data Warehouse
Data and Client Identifications
Querying & Info. Retrieval Tools

Figure 6.5 – The Architecture of the Server Agent

The following sections explore the design in more detail: Representation Tools, aiding users visualizing the concept map; Concept Maps and Views, how this visualization is represented within the workgroup; and the Database Architecture, explains how information is kept.

6.3 Representation Tools

Figures 6.1 and 6.2 show example concept maps drawn with Comet-KO. Nodes, associations and groups are visualized in these examples. The process of creation of concept maps uses tools supported by the client agent. The presentation tools are used in order to generate a private view of the concept map if the map already exists in the workgroup data store. The Map Creation tools aid user to access concept mapping capabilities of Comet-KO. If styles and data types are used or created, Data Type Manipulation tools and Style Creation tools are used. Server side also involves at this operation. Identifications are controlled in order to justify changes done by the user, and global data warehouse may be used in order to store the concept map for further use. All these interactions result in the representation of a concept map. In this section, the visual abilities of Comet-KO, the concept mapping tools, the user-interface interactions, and actions taken on concept maps will be presented.

6.3.1 Concept Mapping Tools

Concept maps are visual languages that are used to represent and understand thought. The concept maps use graph-like diagrams, where nodes symbolize concepts and arcs represent relations between concepts. [25].

A concept mapping language must support three essential features. These are:

- Concepts (data nodes) that symbolize thought and information
- Associations (edges) between concepts.
- Contexts (groups), which contain a group of concepts that are tightly related, and thus extend the meaning of separate concepts. [26]

These essential properties are all implemented in the Comet-KO architecture. Several concept-mapping tools are devised in order to represent and visualize these features.

6.3.1.1 Nodes

Nodes are one of the map creation tools of Comet-KO. Nodes represent the information pieces, namely concepts on a concept map. In a computer-oriented concept-mapping tool, this information could be the contents of a local file, an image, a URL (Uniform Resource Locators), or text that is completely created by the user.

In Comet-KO, nodes have many properties. Any file or URL can be bound to a node. Shapes and images can be used as representations of information. All nodes and their views have a resizable interface for generating different representations to concepts with respect to their importance and/or value to the user. Nodes also have titles and text, editable by the user, which denote the information within the node. These notes and descriptions can be replaced with the visual representation of the node within the map, thus integrating the map and user point of view with the additional information for better understanding of concepts in the map. The text written by the user can be copied either to another node or to any text processor. Cut-Copy-Paste support for text importing and exporting is another feature that eases the work done by the user.

6.3.1.2 Visualization of Information

In a concept-mapping tool, the representation of information should be flexible. If required, user should have the ability to display the content within the concept instead of a representation. All nodes support three types of views. These views are a descriptive icon that can be a shape or any imported image, the concept itself - though this support is only for data in image format now- or text that is written inside of a node. Other types of information like a movie file or any other multimedia documents can also be bound and represented on a concept map via styles bound to particular types of information. These details will be discussed in section 6.3.2.

6.3.1.3 Associations

Associating ideas is an important aspect of any concept-mapping tool. Associations can be directionless, uni-directional, or bi-directional. In addition, they can be between an arbitrary number of objects.

In Comet-KO, associations are represented as edges between any two objects. On the map, each association is represented as one or a set of consecutive line/s between these two objects. A set of lines constitutes orthogonal drawing of an edge [gd-constraints]. While orthogonal edge drawing is supported at the data structure level, the implementation at this state does not support such a mechanism.

An association can be between two nodes or even between other associations. Comet-KO supports Hypergraph drawing for this purpose. In addition, edges, like nodes, also have descriptive properties including a title and textual descriptions. The relation can be further explained using these properties.

6.3.1.4 Grouping

Concept maps are used to organize thought in a more meaningful and understandable manner. However, as Novak [1] stated, as users become more professional in constructing concept maps, the amount of related information grows incredibly large. Maps constructed by organizations may have hundreds of ideas on a single concept map. As the map grows larger, both readability and extensibility of the map decreases substantially.

Any paper or book requires hundreds of pages of notes in order to present and support the ideas within. Take this thesis work for example. Tens of pages are used in order to present a tool based on educational theories, and to support and expand the ideas behind it. As a concept map, this work can be seen as a map with 50 to 100 concepts that are associated with each other in order to present a single context. If the work here can be transformed into a concept map, the work might be summarized within two or three pages. The ratio is huge in terms of volume. But they are both the same in terms of concepts. Thus, reading it constitutes to scanning all the concepts in each form. Another notable issue is that, as we can see, the ideas are collected into seven chapters in a note taking style of work, where a group of tightly related concepts are presented one-by-one within a chapter. Books have contents sections, where reader can have the general idea of the organization and information within the book. The concept maps with only nodes and associations lack this attribute. This ability should also be imported into concept maps.

Although the idea of groups does not appear in most of the works about concept maps, both in computer and classroom based platforms, some information within a concept map are related with each other in a way that, collecting them together does not disturb the context. In contrary, such a way of organizing thought may guide the readers in a way that they can understand the development strategies of the context, and determine where to begin reading for having a general idea. Thus, grouping related information can serve better understanding of ideas and the meaning of a concept map.

Groups should have the same properties a node has. Each group should have descriptive properties as well as their own relations with other members of the concept map, in addition to the associations with their members. These properties are also supported by the Comet-KO architecture.

Some concepts can belong to more than one group in real life. For example, on a concept map about animals, a horse can be categorized as an animal with four legs as well as an animal with a tail. However, a good visual representation of such a condition is very problematic. Thus the property of multiple grouping is omitted and the design accepts that each concept can at most have only one category at a time.

6.3.1.5 Multiple Grouping Views

For sake of better view of the concept map, groups have two different views: An expanded view and a collapsed view. In expanded mode, the group is represented as a boxed region around the members of the group. Each member of the group is shown on the workspace with its relations (edges) with other concepts.

In the collapsed mode, group representation is collapsed into a descriptive image icon. All members within the group become invisible, while all associations of the members of the group and an outside concept is represented as a meta-association between the group representation and the outside concept. One important thing is that these meta-associations are only shown because there is some sort of logical relation between the outside concept and the whole group. However, the descriptive properties of this meta-association are left blank, since the meta-association does not fully reflect the intended relation.

6.3.2 Additional Tools

In addition to supporting the essential attributes of a concept map, Comet-KO provides auxiliary tools to aid user in the knowledge construction process.

6.3.2.1 Cut-Copy-Paste Support

One useful tool in map drawing is to get the information from one concept map to another with some efficient method that does not change the relational organization of the information. Cut-Copy-Paste mechanism is one of the map creation tools of Comet-KO. Traditional cut-copy-paste mechanism is enhanced and applied for this purpose. The selected group of concepts can be cut with (but only with) their inside relations (edges) and can be reapplied to any concept map. Only concepts (nodes) and group of concepts (groups) are in the scope of this mechanism. Associations cannot be copied alone since no associations can exist without concepts.

6.3.2.2 Stored Actions

Each action done on a concept map is identified and stored as action objects on the Comet-KO architecture. The benefit of such implementation is twofold. First of all, stored actions provide trainees to rollback the actions that are done on a map, as an undo/redo system. So, each state of knowledge construction is reachable. Another benefit, which is unimplemented at this state, is that the stored actions provide evaluators invaluable information about the knowledge construction development. The evaluator can watch and examine, and even simulate the whole process by following the actions taken by the trainee.

Actions are stored as follows. Each action object has two properties, the reference to the object that is under the influence of the action and the action index. Many objects can be affected by an action, for example changing the positions of a group of objects on the map. For all affected objects, an action object is created and the information about the action is stored. The index field is used to identify the group of objects affected by this specific action.

6.3.2.3 Styles

As stated above, related concepts can be grouped together. However, consider the case of a concept map that depicts the musical productivity of an artist. The concept groups are constructed on the time periods where the artist's work has substantially changed. The concepts can be the songs and poems written by the artist, where different poems belong to different groups. Whichever groups they belong to, all poems are still a part of a single perceptual group; they are all poems.

The grouping of concepts in a concept map provides support for simplifying thoughts and generalizing the context, though it is highly possible that in a concept map, some concepts may share common properties, which must be visualized in exactly the same way. Styles provide such support. New Styles can be added directly by user and all styles are open to reuse both by the creator and others. Each node, edge and group is supported by a default visual representation at creation time. However, if user desires, he/she can bind a new or a different style to the map element. This is also true for imported files. When a type of file is first added to a map of a workgroup, it has a default visual representation, which is also the default shape of any node. The user can bind a style to the files' mask, so that afterwards every time a file of that type is added to a map in the system, the system directly identifies the association between the style and the mask of the file and displays the style appropriately. In other words, styles also guide the system for visual representation of file-bound nodes.

6.3.2.4 External Application Support

Nodes have three types of visual representations. They can be viewed as a text, image or a shape. What about a movie that is bound to a concept? Today there are hundreds of software and file types on market. It is nearly impossible to write an API that can handle any type of hypermedia sources in market. As new hypermedia software and different types of files are published day by day, it surely is a better idea if users are left to their preferences while viewing a file.

As style support, Comet-KO also provides mechanisms to generate node representation of files, which are launched by different software. At the database level, the information about the software used to launch the specific file is held locally for each user. The user has the ability to change the software used to view the specific file type within Comet-KO. Each entry represents a file type with a specific mask and the software that will launch it. In this way any sort of file can be imported into the concept map, and viewed without making additions or changes to the architecture.

6.4 Concept Maps and Views

While designing the user-to-user interactions, we have to consider different possible interaction models. One possibility is that the trainees work within a fully collaborative environment where the workspace is bound to a single screen for all users and all information construction activity takes place in that single workspace. Each user has the same view of the construct, while all or some portions –this portion can be as small as a node or as big as the whole map– can be updated by only one user at a time. In this workspace, each action taken by a trainee is broadcasted to all other users. So, information consistency is provided for each client's view.

Another model of interaction is that there could be distinct workspaces for each user. Each trainee has his or her own view of a concept map while working. No updates are handled when a change is made on any of the views since each user has his/her own view. Also, as a direct result of this, each user has the capability of reshaping the map at any time he/she desires. The problem with this method is that, after a collaborative session there will be multiple views of a concept map, which are different both in style and content. This means that the concept map is duplicated after a group session.

Both alternatives have their own advantages and disadvantages. From an educational viewpoint, while the first environment is more collaborative than the second, the second alternative promotes individual decisions and intellectual capability to a greater degree in the theoretical case study. Technically however, the single view model is space efficient, while the multiple view model is time efficient. In the single view model, each concept map has only one representation, which is shared among all users, though updates throughout the network are required after each user action on all representation of the concept map. The multiple view model does not require such frequent update, since each user has a private workspace, though, each view must be stored respectively.

As R.F. Dugan [36] clearly stated that in highly graphical applications, such as a concept-mapping tool, the performance and data consistency could be great limitations on the application. He also stated that simultaneous updates on each user's machine could produce unacceptable runtime results due to unreliable network performance, and lead to improper share of resources between peers.

Moreover, as discussed earlier, knowledge construction process requires both a collaborative phase and an individual phase. In the collaborative phase, past knowledge and interpretations of the content are shared while in the individual phase the gathered ideas are evaluated and criticized according to one's cognitive structure. This shows that the software should both support individual and collaborative working. More than this, these phases should be related. Once user gathers ideas in his/her mind in an individual process, he/she needs to express these ideas with others. After sharing and gathering the external information, user must process these ideas, find the most appropriate representation of them and construct the concept map accordingly; and all of these procedures must be done, again individually.

Obviously, the design should take these ideas into consideration. Thus, the intention is not to have a fully dynamic workspace among users. That is, in this implementation, the updates are not directly reflected to the collaborators. Rather, global and individual maps are stored together as duplicates in a global workspace, and each user has access to them.

6.5 The Database Architecture

The goal of the Comet-KO architecture is to have a distributed system that enables collaboration with other users in a workgroup. The most important point about the database architecture is that, each workgroup should have unique user identifications, by which each session can identify its users and collaborate with them. These identifications hold the authority levels of each user, where authority levels regulate the access rights of clients to a specific map. The owner of the server session can adjust the authority levels. In addition to this, if required, each client should have the ability to work alone. This led to a client-server architecture where a server is connected to many clients at a time, serving them with distinct identifications and data. The server also has the duty to warn users about the updates on the concept maps and, if requested, to update their workspaces accordingly.

6.5.1 Database Organization

The Comet-KO database consists of ten tables, which holds information about the workgroups, users, accessibility rights, authority levels, multiple views of maps within these workgroups, and contents of the map elements. Moreover, it stores information about previously created re-usable styles and how the client machine handles special requests from a node related to a hypermedia document. These tables are "Files", "Native Nodes", "Concept Map", "Map View", "Object" "Association Rules", "External Application", "User Identifications", "Styles", and "Workgroup Identifications". The relations of map elements and external information sources will be presented later in this chapter. The entity relation diagram of the Comet-KO database is presented in figure 6.6.

- **The Files Table**: This table holds references to any external source bound to a node in a concept map. The content type (MIME) (mask) is needed in order to determine which application is needed in order to present the specific file. The exact address of the file is needed in order to receive the data from that location.

- **Native Elements Table**: Though the user has the ability to use data created in other applications as map elements, he/she also has the ability to create map elements not associated with any files, like links, arrows, groups and even nodes that does not refer to any file. These non-data elements of a concept maps can collected under the name of native elements. Information about these elements is stored in the Native Elements table. The name Native Elements table is used for any object in this table has no association with any file and they are created in order to explain associations and relationships of objects and pointing out the crucial points on the concept map.

- **Concept Map**: Concept map table is used for identification of the different concept maps residing on the global storage area.

- **Map View Table**: In Comet-KO, each user may have different views of a concept map. "Map View" table relates each user's view to the corresponding concept map in the workgroup data store.

- **Objects Table**: Both elements from the Files Table, and the Native Nodes Table are used as map elements. They also have some common attributes including coordinates, representation styles, etc. The Object table is used for mapping the data both in the Native Elements, and the Files tables to a common table. Since all visuals representation data is stored in the Objects table, the presentation specific queries only run over the Objects Table.

- **Association Rules**: Each element within a Comet-KO map may have relations with other elements. Association Rules maps these relations.

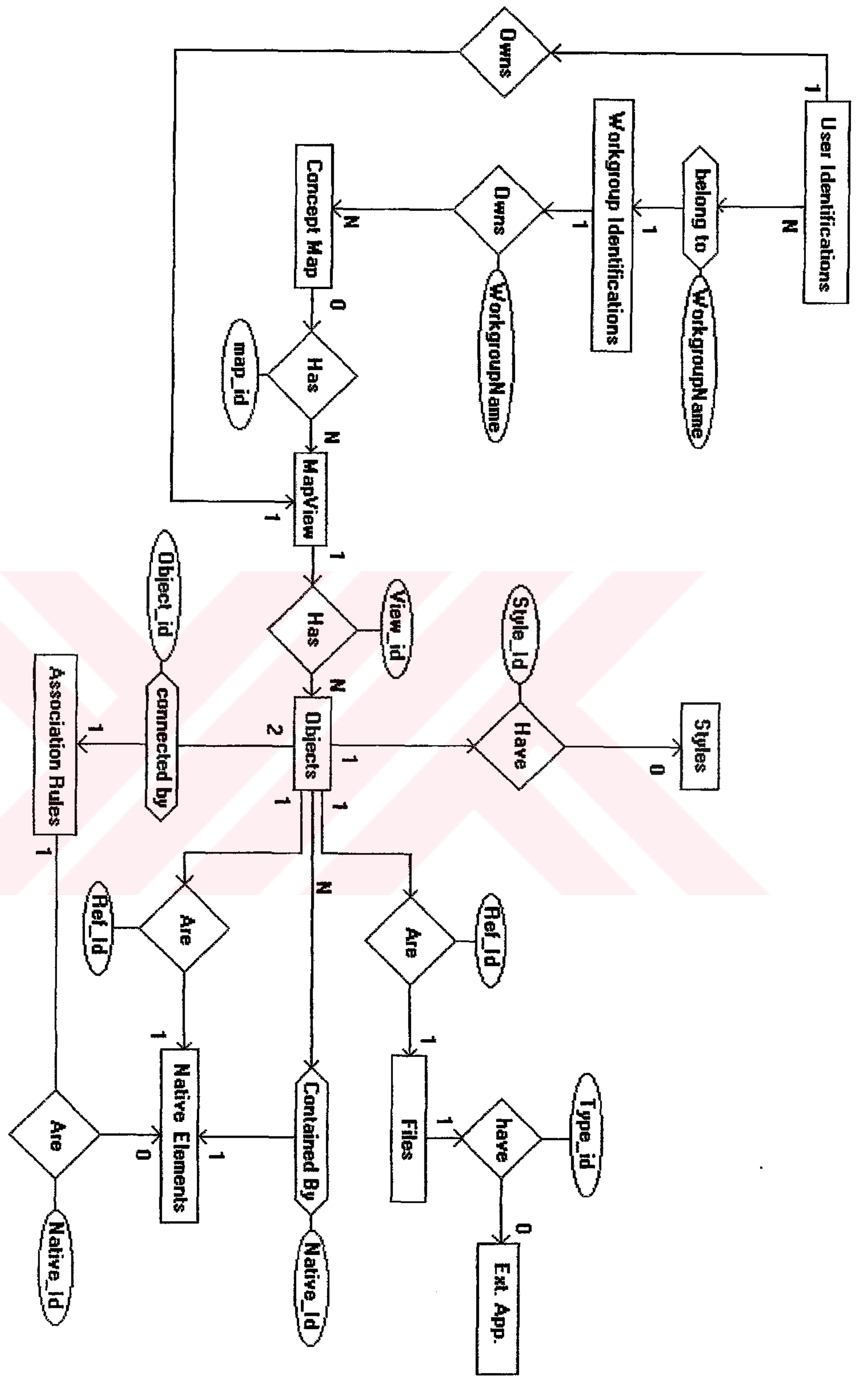


Figure 6.6 – The E-R Diagram of the Comet-KO Database

- **Styles Table**: In Comet-KO, the user can change visual appearance of Comet-KO objects by giving them different shapes, color and designs. The user can create these new styles for each object and can use these newly created styles in the future. The styles database table collects these created styles, associating them with the user who created them.

- **External Application Table**: This table associates a file type and its content type (MIME) with a specific application program. By means of this table, users can view any node bound to a file.

- **User Identifications**: This table holds the data about the clients. The authority is also implemented in the database at this table. Each user has a specific authority level entry for each workgroup and all the maps within the workgroup.

- **Workgroup Identifications**: Since the architecture presented here has a single server for large number of clients, it is probable that more than one group may use the same server for different purposes. Hence, identifications of the workgroups are needed as well as identifications of the users. The Workgroup Identifications table holds both descriptive and authorization information about these workgroups.

The Comet-KO database can be separated into two parts with respect to access. Most of the information, which are the workgroup, and user and map related information could be accessed both locally and remotely. However, two of the tables, that is, styles and external applications tables can only be accessed via the local machine. The reasons and motivations will be presented in this chapter while presenting the design of Comet-KO.

In this architecture, user identifications and workgroup identifications represent the rights of the users over a group of concept maps. Some users may have the right only to view a map while some may have rights to alter the concept map. This property also enables evaluators to focus on a single trainee at a desired time, and to watch and shape the information construction activity whenever appropriate. More than that, with suitable access rights, one can lock some portion or all of a concept map for some period, informing others that some work is going on with the locked elements, and limit outside changes that will be done on these subjects in that period. This could also be a useful tool for evaluators, such that, by using this capability, evaluator can support the trainees with unfinished information that is left to the users in order to fill in the gaps, and thus, shape the course of the session.

6.6 Implementation

6.6.1 Details

The Comet-KO design consists of 51 classes and more than 25000 lines of code. Most of these classes are used in order to store and manipulate information supplied by the users. In addition to these classes, many visual and non-visual component classes are retrieved and used from the reusable component library of Borland C++ Builder 4.0. In this section, these derived components will not be presented. For more information about these components refer to [34].

In this section the class hierarchies, their interactive corporation, performance issues and some critical actions handled within a session will be presented. The UML diagram of Comet-KO can be seen in figure 6.7. The starting point for the application is the "Main" class, which is derived from the "TForm" component and initiated as a SDI (Single Document Interface) window. The main class controls the interactions between several concept maps, and supports a buffer for global cut-copy-paste operations that can combine several map constructs. These actions are invoked by accessing the active map window's information using its routines. This relation is not shown on the class diagram for sake of simplicity. The "Main" class also handles interactions with the database.

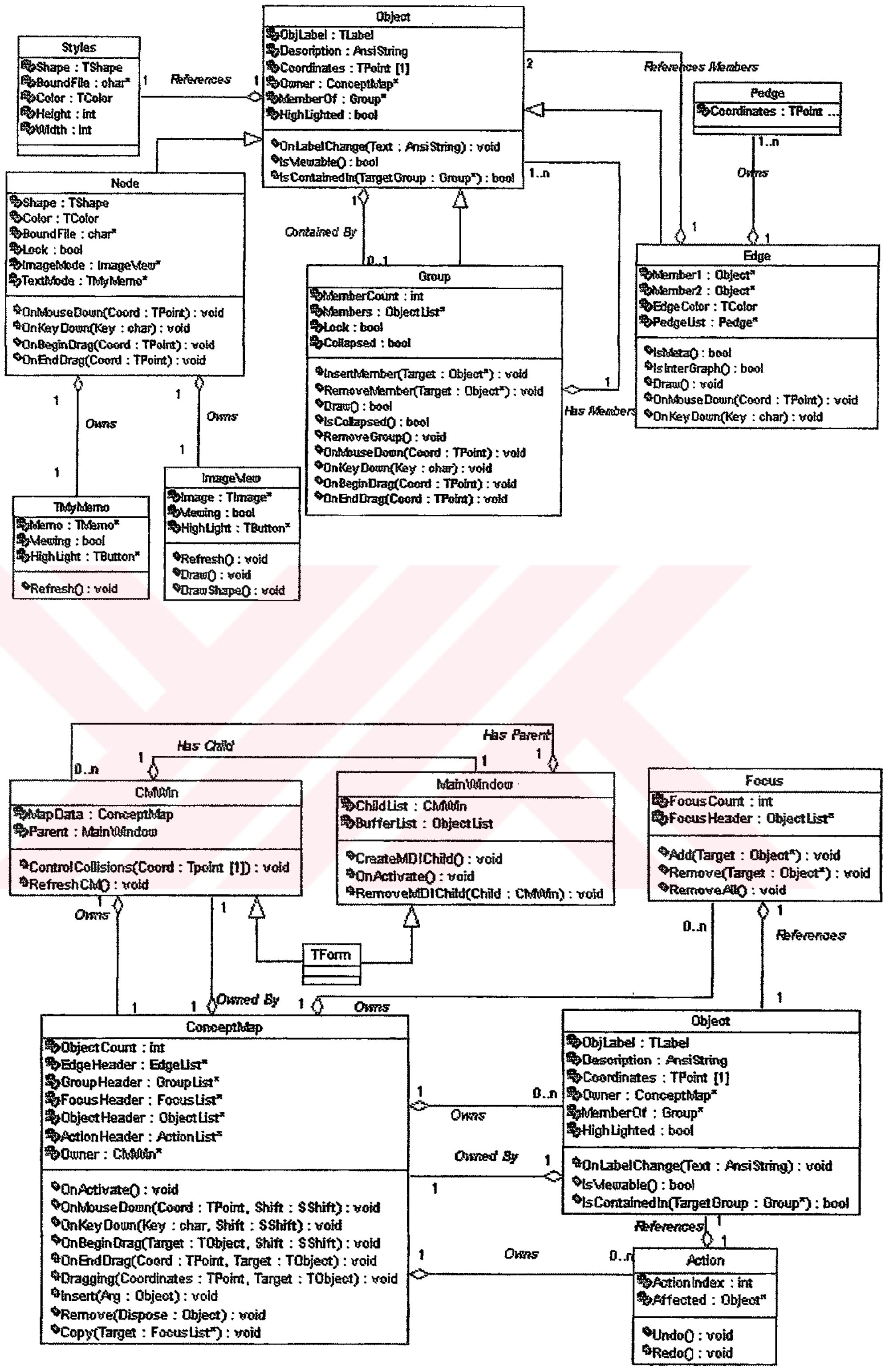


Figure 6.7 – UML Diagram of Comet-KO

The “Main” class has the ability to initialize “ConceptMapWindow” instances. The “ConceptMapWindow” is also derived from the “TForm” component and initiated as a MDI (Multiple Document Interface) window. This enables user to have more than one concept map working at any time. Each ConceptMapWindow object has “has parent-has child” relations with its owner class “Main”. The ConceptMapWindow class is also responsible for the graphical actions on itself. It owns routines for refreshing the map in various ways.

The ConceptMapWindow instances have routines for refreshing itself on the screen as well as routines for detecting the changed areas and refreshing only these specific regions. This ability is supplied by the “ControlCollisions” method. The “ControlCollisions” method is a performance-critical method. It takes a square region as input. On this square area it determines the objects affected by a specific action and invoke their refresh methods. In this way, not the entire region, but only the affected objects within the effected region are refreshed.

The ConceptMapWindow also supplies a couple of event handlers. These event handlers are invoked when a user takes an action related to the map itself. Example handlers are, “BeginDrag” handler for starting a drag operation or “MouseDown” handler when user clicks on the map. The Event handlers are built-in properties of the “TForm” component, which supply an interface for the actual handles.

Each ConceptMapWindow creates a “ConceptMap” instance at construction time. The ConceptMap class holds logical information about the map. It handles logical operations like copying objects, inserting new elements to the map and deleting the elements from the map. The objects that are deleted are only logically removed from the concept map. This is because the deleted objects may be needed later for an undo operation.

Each concept map has five main attributes: object list, edge list, group list, action list and focus list. The object list consists of a doubly linked list for logical storage for objects. The edge and group lists are a generalization of the object list, which are also linked lists referencing the elements in the object listing. Note that there is no separate list for nodes. The reason behind this is that there are multiple actions that only affect the groups. For example after a move action of a group member, domino effects of the groups must be regulated. Also the same is true for the edges since the graph structure supports hyper graphs. Upon a move action, the edge list should be controlled many times for each edge that is between other associations. However, there is no action that only affects the nodes.

The focus listing represents the set of entities currently selected by the user. Each map has its own focus, and as a result, the user working on more than one window does not lose the state at each map via switching through another. The focus list is also a reference to the object list. All these list classes have the responsibility to logically fetch, insert and remove requested entities.

The "Object" class is the base class for each entity that can be placed on the map. The descriptive abilities and place of the object is held inside the Object class. The descriptive abilities are stored in different built-in components. "Tlabel" is the component used to store and display the title of the object while an "AnsiString" class is used to hold any text within TLabel. The object class also initiates a reference to styles, so that the created objects can inherit attributes of their style types.

Three classes are derived from the objects. These are edges, groups and nodes. Each of them is a specialized type of map entities and has distinct attributes.

The Edges are visual representations of associations between objects. At this level, each object has the ability to draw itself on the map. Edges are also responsible for determining whether they are inter-group edges, which are between two members of the same group, or meta-edges, which are between a member of the group and an outside object. This logical property of the edge is used when the edge draws itself on the screen. The edges support orthogonal edge drawing at the data structure level. Each Edge object references a list of "PEdge" class objects. Each "PEdge" object is a basic implementation that points out the coordinates of a portion of the edge. For simplicity, this relation is also omitted in the class diagram.

The Edges also hold references to the members of the association. Each edge has exactly two members. These members can be any objects, thus the data structure also supports hypergraph drawing. The maintenance of hypergraphs is a hard task, which degrades the performance considerably. When any member of any edge moves, the edge listing should be controlled iteratively many times until all edges are controlled for a move. If any hypergraph edges are found which are affected by the previously affected edges, then all such edges should also be redrawn. This algorithm does not use any heuristic methods in order to stop the iterative process and is deadlock free. Appendix I gives the algorithm and a proof that this algorithm is deadlock free. The number of iterations is solely based on the size and complexity of the graph, so there is no way to limit the iterations. However, for better performance, we use the definition of an edge. "An edge is an association between two objects". Hence, no edge can exist without its members. So, by this definition we can say that if the edges are sorted with respect to their creation order, which is the case, then after each iteration, the older edges than the last updated edge cannot be affected from any other edge move. By this observation although the number of iterations cannot be reduced, the search depth is reduced after each iteration.

"Groups" is another specialized class derived from the object class. It has a collapsed representation, and a list of members, which are again any objects in the map. The Groups class is responsible for drawing itself on the screen. It also has the responsibility to maintain logical operations done on the group members. Examples of these operations are adding or removing a member, invoking move actions that directly targets the whole group and etc.

"Nodes" represents concepts in the map. The "Nodes" class is also derived from the "Objects" class. Each node has two visual components. A "TextualImage", and a "ContextImage". These components are initialized as instances of "TMyMemo" object, and "ImageObject" object respectively. Each "Node" object also reference to its owner groups and can invoke domino effects of the specified group.

Both “TMyMemo” and “ImageObject” are derived from “TMemo” and “TImage” classes respectively, which are derived from built-in component library of C++ Builder. These details are also omitted on the UML diagram. Each class has visual abilities, as well as shared event handlers. The visual abilities are resizing of the visual component, controlling the visible properties of their representation, and refreshing, redrawing and clearing themselves. These two visual classes also have handlers for the “BeginDrag”, “MouseDown” and “KeyDown” events.

As explained above, there are two separate groups of event handlers in this implementation that regulate the actions taken by the user. One group of event handlers is attached to visible components, namely nodes and groups. The other group of event handlers is attached to the ConceptMapWindow instance. The handlers on the nodes and groups generate mouse down, key down, resize, begin drag, end drag, accept drag, and dragging events of their owners, while the handlers on the mapping window are used to engage in mouse down, key down, and accept drag events of both edges and the window itself. There are two reasons behind such an implementation. First of all, nodes and groups are interactive elements of a concept map, while edges are not. User can move, resize or move any node or group, though edges can only change state when one of their members is updated. Adding event handlers to edges will only be a burden to the system. On the other hand, using only one event handler for all elements in a concept map will surely decrease the complexity of the code both in the aspect of readability and performance. Experiences with concept maps [Novak] show that, the biggest concept maps used by organizations today do not have more than 200 nodes (concepts). Our experiences also justify that, for such numbers, the event handlers at each node do not cause ill performance results, while splitting the event handlers improves readability of the code. Moreover, splitting handlers into two separate groups means, they do not need to search the whole Object list to distinguish which Object group/s is/are selected, so, improving the performance.

6.6.2 Actions

An action is any atomic interaction between the user and the concept map. When the user interacts with the map, for each element of the concept map that is affected by the action, an Action object is initialized. This Action object has the responsibility of finishing the requested action as well as adding itself to the action list. This list is used to provide users with undo and redo facilities.

6.6.2.1 Action Classes

All actions are derived from a base "Action" class. The base class provides an interface for all derived action classes as well as a reference to the affected object of the workspace and a shared index for the action itself. The action objects are stored in each map window, and handles both doing and undoing of the specified action.

An action is created when the user interacts with the map window and alters the context or visualization of the map. At that time, firstly, an action object is initialized for each affected object. This object is inserted into the action listing of the map window for further referencing. Then the "Redo" method of the specific action object is called in order to reflect the changes on the window.

When an undo operation is called, the window determines the index number that is last given to the action. By referencing this index, the action list is traced and "Undo" method of the action object is called.

6.6.2.2 Actions

Eleven actions are identified in Comet-KO. These actions are:

- Add action: adds any kind of map object to the concept map. The reverse of this operation is deleting the referenced object.

- Delete action: deletes any kind of map object from the concept map. The reverse of this operation is adding the object back to the map. Note that for undoing a delete operation, the deleted instance is required. In order to supply this feature, it is assured that no object is physically deleted after such operation; the specified object is only removed from the map and its references. The objects are deleted from the memory (C++ does not support garbage collection) only when the map window is closed.
- Move action
- Resize action
- Group action
- Ungroup action: which is different from deleting a group. When deleting a group, all members of the group are also deleted, while ungrouping only removes logical grouping.
- Shape change action: also constitutes to binding an image to a node.
- Title Change action
- Description Change action
- Color Change action, which is only applicable if the related object is a node and it is represented as an icon.
- Mode Change action, constitutes to changing the visual representation of a node or a group. The mode change in nodes constitutes to changing the icon representation to a text view. The mode change in group objects constitutes to collapsing and expanding the groups. The mode change action in edges constitutes to changing the edge's directional properties (e.g. transforming a uni-directional edge to a bi-directional edge.).

This action list covers the complete capabilities of a user interacting with the interface. The only action that is omitted, thus removed from listing is changing the bound file of a map node. Since a node is created with its concept (information) binding new information to an already created node constitutes to removing the object and inserting a new object with a new concept.

Each of these actions has separate undo and redo implementations, which are invoked just after the action is initialized or directly when an undo/redo operation is initiated. These undo and redo methods invoke the event handlers of the window and/or the objects.

Chapter 7

Conclusion

This thesis presents Comet-KO, a knowledge organization tool that enables users to collect and, visualize information using concept mapping techniques. In Comet-KO, nodes representing concepts or ideas can be visualized in a variety of graphical styles, including images and shapes, and a memo representation for text. Edges constitute the relationships between ideas (nodes). A set of possible edge drawing methods such as orthogonal edge drawing and hypergraph drawing are supported in order to facilitate the presentation of such associations.

A grouping mechanism is also implemented as an extension to conventional concept mapping languages. With the grouping mechanism, a user can represent contexts; a group of concepts that are related with each other. This proves to be a powerful additional feature capability. Comet-KO also provides support for styles, giving the user a variety of tools to organize knowledge in a more clear and understandable way.

Comet-KO is based on instructivist and social constructivist learning theories. In this respect, as a computer-mediated educational tool, it meets the requirements of both teacher-centered instructivist pedagogic theories and student-centered creative work.

Performance is one of the main criterion in the implementation of Comet-KO. Considerable effort has gone into the selection of algorithms and data structures designed to speed up refresh and fetch operations.

At this state, Comet-KO is not fully implemented. Although the visualization part and tool to access the database are completed, they are not connected to each other.

Two auxiliary tools that are not implemented at this stage may have great positive impacts on knowledge construction process by Comet-KO. The first tool is a map development simulator, by which evaluators could track the development of a specific concept map. By not only seeing the final product, but also observing the development process, evaluators can judge trainees in a more meaningful and objective way. The basic implementation of such a tool requires structures and functions for capturing each action taken by the user. These structures and functionality are already implemented as action classes that are used for undo/redo option. Consideration of how best to analyze and present this information may be an interesting topic for further research.

Another, possible tool is an organizational tool that designs the layout of a specific concept map automatically. If implemented carefully, with such a tool, users could devote more of their time to thinking about the material in a map, and be less concerned about its visual representation. This will lead to more solid and creatively constructed concept maps, which is the aim of Comet-KO as an educational software.

Bibliography

[1] Joseph D. Novak. Learning, Creating and using knowledge: Concept maps as facilitative tools in schools and corporations. Lawrence Erlbaum Associates, Mahwah New Jersey, 1998.

[2] Howard Gardner. *Multiple intelligences: The theory in practice*. Basic Books, New York, 1993.

[3] Elizabeth Duffrin. *Direct instruction making waves.* Catalyst: Voices of Chicago School Reform, September 1996.

[4] Jeanne Allen (ed.). *A Nation at risk: An educational manifesto*. The Center for Educational Reform, Washington DC, 30 April 1998.

[5] Mark Schug, Sara Tarver and Richard Western. *Direct instruction and the teaching of early reading*. Wisconsin Policy Research Institute Report, Vol. 14, No. 2, March 2001.

[6] Susan Hanley. *On constructivism.* Maryland Collaborative for Teacher Preparation http://www.towson.edu/csme/mctp/Essays/Constructivism.txt

[7] Charles Crook. *Computers and the collaborative experience of learning.* Routledge, London, 1996.

[8] University of Massachusetts Physics Education Research Group. *A Constructivist view of science education.* http://umperg.physics.umass.edu/

[9] David H. Jonassen, Kyle L. Peck, Brent G. Wilson. *Learning with Technology: A Constructivist Perspective.* Prentice Hall, 1999.

[10] M.B. Tinzmann, B.F. Jones, T.F. Fennimore, J. Bakker, C. Fine, J. Pierce. *What Is the Collaborative Classroom?* NCREL, Oak Brook, 1990.

[11] G. Salomon. *On the Nature of pedagogic computer tools. The case of the wiring partner.* In S.P. Lajoie and S. J. Derry (ed.), Computers as cognitive tools. Lawrence Erlbaum Associates, Hillside, New Jersey, 1993.

[13] Virginia Richardson. *Constructivist teacher education: Building a world of new understandings.* The Falmer Press, Bristol, 1997.

[14] Linda Roberts. *The Internet in the classroom.* Harward Conference on the Internet and Socitety, may 29-31 1996.

[15] J. Feldhusen, M. Szabo. The advent of the educational heart transplant, computer-assisted instruction: A review of the research. Contemporary Education, 40, 265-274, 1969.

[16] A. Bork. *Learning Through graphics.* In R. Taylor (ed.) The computer in the school: tutor, tool, tutee. New York, Teachers College Press, 1980.

[17] D. Sleeman, J. Brown. *Intelligent tutoring Systems.* New York, Academic Press, 1982.

[18] Chris Dede, Marilyn C. Salzman, R. Bowen Loftin. *Sciencespace: Virtual realities for learning complext and abstract scientific concepts.* Proceedings of the IEEE, Virtual Reality Annual International Symposium, p. 246-252, 1996.

[19] A.I Wars Official Home Page. http://www.tacticalneuronics.com/ai.htm

[21] Sandy Britain, Oleg Liber. *A Framework for Pedagogical Evaluation of Virtual Learning Environments*. JTAP Report No 41, 2000.

[22] The Narrative Immersive Constructionist/Collaborative Environments project. University of Illinois, Chicago. http://www.evl.uic.edu

[23] IBM Lotus Software, *LearningSpace*: http://www.lotus.com/products/learnspace.nsf/wdocs/homepage

[24] Tony Buzan. *The Mind Map Book*. BCA, London, New York, 1990.

[25] A. Huff. *Mapping Strategic Thought*. John Wiley & Sons Ltd. 1990.

[26] Luis M. Carrico, Pedro M. Antunes, Nuno M. Guimaraes. *Visual Reflection: Language, action and feedback*. IEEE Symposium on Visual Languages, Tokyo, Japan. September 13 - 16, 1999.

[27] Peter Russel. *The Brain Book*. Routledge, London, 1990.

[28] Martin Owen, Mario Barajas. *Implementing virtual learning environments: Looking for holistic approach*. Educational Technology & Society 3 (3) 2000, ISSN 1436-4522.

[29] David H. Jonassen. *Computers in the classroom: Mindtools for critical thinking*. Prentice Hall, Englewood Cliffs, New Jersey, 1996.

[30] IHMC, Institute for human and Machine Cognition, University of West Florida. *Constructivism*, http://www.coginst.uwf.edu/

[31] Brian L. Gaines, Mildred L. G. Shaw. *Collaboration Through Concept Maps*. Proceedings of CSCL95: Computer Supported Cooperative Learning. Bloomington 1995.

[32] Romain Zeiliger. *Concept-Map Based Navigation in Educational Hypermedia : a Case Study*. Proceedings of ED-MEDIA'96, Boston, USA, 1996.

[33] Ozden Emek Erarslan. *Rememex: A Software tool for knowledge construction and organization.* M. S. thesis, Bilkent University, Department of Computer Engineering and Information Science, 1998.

[34] Charlie Calvert. *Borland C++ Builder Unleashed.* Macmillan Computer Publishing, Indianapolis,USA, 1997.

[35] Thomas A. Funkhouser. *Network topologies for scalable multi-user environments.* Proceedings of the IEEE, Virtual Reality Annual International Symposium, p. 222-228, 1996.

[36] Robert F. Dugan. *An architecture for testing synchronous multiuser software.* Ph.D thesis, Department of Computer Science, Renselaer Polytechnic Institute, 1998.

Appendix I

Definition of the refresh algorithm:

Edges can be between any objects in the map. As a result, the concept map is a hypergraph. In this graph, edges themselves cannot be moved, but updated only when one of their members is affected by a move action. When any action is taken that cause updates on the state of the edges, the update operation is handled as follows:

1. At the first step, only edges that are directly connected to only nodes and groups are updated.
2. After this operation, the edge list is iteratively searched for any hypergraph edges that contain any of the updated edges as a member. If any hypergraph edge is found, it is also updated.
3. The edge list is searched for any hypergraph edges that contain one of the edges that are updated at this iteration. If any hypergraph edge is found, it is also updated.
4. Stage 3 is repeated until no edges require an update.

Theorem: The hypergraph refresh algorithm presented above is deadlock free.

Definitions:

G is a graph G=(V,E) where V is the vertex set and E is the edge set of the graph. Let M= $\{m_1, m_2, \ldots, m_k\}$ be a set that maps each vertex and edge on the same set. An edge E=$\{m_1, m_2\}$ is drawn between exactly two members of set M and m_1, and m_2 are its members.

An *update path* is a set of edges P=$\{E_1,E_2,\ldots,E_n\}$ where:

$E_1=\{m_1, m_2\}$

$E_2=\{m_3: E_1, m_4\}$

$E_3=\{m_5: E_2, m_6\}$

.

.

.

An *update cycle* is a set of edges C=$\{E_1, E_2, \ldots.E_x, E_1\}$ where:

$E_1=\{m_1, m_2\}$

$E_2=\{m_3: E_1, m_4\}$

$E_3=\{m_5: E_2, m_6\}$

.

.

$E_x=\{m_a: E_{x-1}, m_b: E_1\}$

$E_1=\{m_1: E_x, m_2\}$

A *deadlock* on this algorithm occurs when algorithm start update operations from one edge and continuously update the edge set. This deadlock requires that updated edges generate an update cycle in the iterative process.

Proposition 1: A deadlock exists in graph G if and only if there is an update cycle in the edge set E

($\Rightarrow$) A deadlock exists if there is an update cycle in edge set E

Figure A.1 illustrates a basic update cycle. Deadlock on this update cycle can be seen clearly.

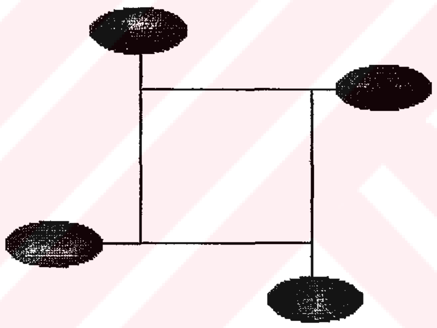


Figure A.1 – A Basic Update Cycle.

($\Leftarrow$) There is an update cycle in edge set E if a deadlock exists.

Proof by contradiction:

Assume that there is a deadlock in graph G where there is no update cycle in edge set E. As there is no update cycle in edge set E, there is an update path between two connected members of M, namely m_1, and m_n, and no update path between any member of this update path and m_1

By definition, this update path can be written as:

$P=\{E_1, E_2, \ldots E_k\}$ where,

$E_1=\{m_1, m_2\}$

$E_2=\{E_1, m_3\}$

$E_3=\{E_2, m_4\}$

.

.

.

$E_k=\{E_{k-1}, m_n\}$, and there is no other edge $E_t=\{m_t, E_1\}$ where E_t exists in path P.

In this case all updates on this update path can be done without a deadlock by definition, which is a contradiction.

Proof of Theorem:

By proposition 1, it is known that there can be a deadlock if and only if there is an update cycle between edges that are updated.

Proof by contradiction on creation time of edges:

As edges can be drawn between exactly two entities in set M, left $E_1=\{m_1, m_2\}$ be the first drawn edge of the update cycle $C= \{E_1, E_2, \ldots E_n, E_1\}$.

Then E_1 can be also written as $E_1=\{E_n, m_1\}$ (As presented it must hold the both conditions $E_1=\{m_1, m_2\}$ as the first element of the set C and $E_1=\{E_n, m_k\}$ as the last element of the set C) though E_n does not exist at the time edgeE_1 is drawn. So by contradiction edge refresh algorithm is deadlock free.