REQUIREMENT ELICITING PROCESS: A METHOD TO ANALYZE REQUIREMENTS THROUGH CONCEPT MAPS

A Paper
Submitted to the Graduate Faculty
of the
North Dakota State University
of Agriculture and Applied Science

By

Marie Nilukshi Fonseka

In Partial Fulfillment
for the Degree of
MASTER OF SCIENCE

Major Department:
Computer Science

May 2014

Fargo, North Dakota

North Dakota State University
Graduate School

Title

REQUIREMENT ELICITING PROCESS: A METHOD TO ANALYZE REQUIREMENTS THROUGH CONCEPT MAPS

By

Marie Nilukshi Fonseka

The Supervisory Committee certifies that this ***disquisition*** complies with North Dakota State University's regulations and meets the accepted standards for the degree of

MASTER OF SCIENCE

SUPERVISORY COMMITTEE:

Gursimran Walia
Chair

Dr.Saeed Salem

Dr.Simone Ludwig

Dr.Limin Zhang

Approved:

05/16/2014	Dr.Ken Magel
Date	Department Chair

ABSTRACT

Conceptual modeling involves the understanding and communication between system analysts and end-users. Concept maps (CM) are informal, semantic, node-link conceptual graphs used to represent knowledge in a variety of applications. Concept maps capture knowledge about the concepts and concept relationships in a domain, using a two-dimensional visually-based representation. In this paper we examine hoe concept maps created for student requirements are different from that of a domain expert. This will aid us as analysts, to understand how well we interpret requirements in known and unknown domains using two requirements eliciting methods problem description and use case method. The results show as that we tend to have more simple and modular requirements in known domain regardless of the method used and have more complex set of requirements in unknown domains.

ACKNOWLEDGEMENTS

This research could not have been accomplished with only my contribution, but there were many individuals who genuinely helped me in various ways. First and foremost, I would like to thank my advisor Dr.Gursimran Walia for accepting me to as a student, and providing me freedom to learn. His guidance, encouragement, patience, thoughtfulness, and genuine support for my research, academic achievements, as well as completing my thesis were enormous. I would also extend my gratitude to Dr.Saeed Salem, Dr.Simone Ludwig and Dr.Limin Zhang for accepting to me on my committee in such short notice. This wouldn't be possible without your help and quick response. And Dr.KenMagel for valuable advice during the course work of my studies.

I would like to give my gratitude to my colleagues at NSDU Computer Science department for their support and collegiality, and staff at the NDSU Graduate school and computer science department, without your help and guidance this wouldn't have been a reality.

Last but not least my heartiest gratitude goes to my parents, and my sister especially my dad for his words of wisdom which held me together in times of uncertainty, for giving me courage to move forward, I owe them for giving me their best throughout my life. I am deeply indebted to my in-laws s for their love and enormous help in various ways. And finally my loving husband Keith Abeyratne and our children Jamie and Kaylee are remembered with my whole heart. Their caring, loving, and supportive environment shadowed and strengthened me throughout all ups and downs during past years. My heartfelt thanks go to my loving children for their love, help, and patience in my absence. You have been my enduring source of love, strength, and encouragement.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

Requirements engineering is a customer centric process, which concerned with the acquisition, analysis, description, and validation of software system. Requirements engineering has many iterative phases and Requirements elicitation is perhaps the most difficult, most critical, most error prone, and most communication-intensive aspect of software development. Requirements analysis involves frequent communication with system users to determine specific feature expectations, resolution of conflict or ambiguity in requirements as demanded by the various users or groups of users. Requirements acquisition is an especially a high hurdle because stakeholders are often unclear about objectives. Misunderstandings of user requirements between software developers and users will cause problems in terms of satisfying user needs, defects, cost and schedule during the software development process. [2] Many of the software problems arise from shortcomings in the way people gather, document, agree on and modify the product's requirements. Most of the problems areas might include informal information gathering, amplified functionality, erroneous and un-communicated assumptions, [1] and possibly the most common problem in the requirements analysis phase is that customers' have only a vague idea of what they need, and it's up to the Analyst to ask the right questions and perform the analysis necessary to turn this amorphous vision into a formally-documented software requirements specification. Often, customers and engineers fail to communicate clearly with each other because they come from different worlds and do not understand technical terms in the same way. This can lead to confusion and severe miscommunication.

The success of any information system depends on how, we as software engineers capture customer requirements. Many misunderstanding of the requirements may occur as the customer’s and the engineer’s understanding of the system to be created can be interpreted differently. This may result in delivering incorrect functionality, unwanted features in the software that the customer may not need, hence may require a lot of rework and wasted resources. The knowledge gap of what needs to be developed and what is ultimately delivered is a crucial factor in making any software system a success. During the initial phase of any system development, software developers are challenged to uncover, understand, and specify the user requirements [3]. Thus, it is important that there should be a common understanding between the users/customers, project managers, and developers and other stakeholders with respect to the requirements of the software system being developed. This is one of the major risk factors in all software projects [4]. The sooner misunderstandings are resolved, better is the ability of developers to build a product that meets user requirements and less is the expense associated with corrections later in the life cycle [5, 6]. To design a software product acceptably the software engineering team has to understand what needs to be done in customer’s aspect. How customer uses various techniques, processes, products and goals to accomplish a given task.

Requirements engineering is recognized as a social process that is characterized by ongoing sense making among participants, which include managers, end-users, and system analysts [16]. Moody et al. [17] claimed that software development is more like a craft than an engineering discipline during the processes of requirements engineering. In this respect, human cognition plays a pivotal role in understanding human and organizational issues in requirements engineering, and in identifying ways to improve the quality of conceptual modeling.

Cognitive mapping techniques have been widely used in strategic management and political science to depict and explore the cognitive structures of members of organizations [19].

Some researchers, such as Avison and Fitzgerald (2003) [18] and Montazemi and

Conrath (1986) [20], have hinted at the usefulness of cognitive mapping in systems development. However, they have only provided brief coverage on the use of cognitive mapping. There are many techniques available to aid us in-order to understand what customer's expectation of the software system is, in this paper we try to analyze customer's mental model regarding the system under construction. Mental model may be used as a guide for the design of the solution. Further it may be used as a tool to make good user and business decisions by focusing the solution on what users actually need – based on the feedback they have already given you.

In this paper, we have provided a comprehensive review of mental models and methods for eliciting and representing mental models. We also discuss how cognitive analysis can be employed to improve the quality of requirements analysis process by understanding what each stake holder is visualizing regarding the system to be implemented, and how they supplement popular systems development methodologies in conceptual modeling.

1.1. Problem Statement

Requirements elicitation is one of the most important stages of systems analysis, as it is at this point that clients and analysts work together to determine the requirements of a new system to be developed. If the system does not meet client's expectations, then the project is essentially a failure. Requirements Elicitation is one of the most difficult stages of analysis, with numerous communication barriers existing between the analyst and client that make eliciting requirements difficult. Analysts and clients often speak in different general languages, with analysts often being more technical in nature, while clients will often speak more from a business perspective. This makes common understanding difficult. In many situations, users are not sure what they are looking for in a new system or product and have trouble articulating their real needs. Requirements elicitation cannot be performed off handedly by simply asking the user to tell the developers what they want. In many cases users are not aware of how software is created and cannot describe their needs in a way that developers can work with. For example: Imagine if someone only had a basic knowledge of the modeling language used to represent a conceptual model of a house and we asked if the house presented in the floor plans, electrical plans and plumbing plans met their needs. Unfortunately system architects ask stakeholders these questions every day. Consider a contractor building houses and only showing clients floor plans. Based on these floor plans the client must agree or disagree that this is the house they need. To compensate for a lack of understanding, stakeholders will trust that the people building the house understood what they wanted and will let them continue building. Once the house is complete only then will the client discover that this truly is an ugly house that does not meet any of their needs. At this point the contractor can then either fix the problem or contend that the client agreed to the floor plan and therefore they got what they deserved.

Would this scenario ever happen in reality? No, because in reality homebuilders not only have floor plans but also have 3-D pictures of what the interior and exterior of the house will look like, a vision. The client can easily use this vision to see if their wants and needs are being met long before anything is built. In other words, the homebuilder and the client have a shared understanding of what is being proposed. Using this shared understanding all stakeholders can collectively move toward the same goal.

When we consider building a software, consider the different artifacts created during the different stages of software system development, the requirements document, the software architecture document, software design document, source code, schedules etc..Requirements documents are typically expressed in natural language, design and architecture documents are often represented in more technical formats like Unified Modeling Language (UML) representations and schedules are often represented using GANTT charts. The reason that there are so many representations for the documentation in a system is that each representation is superior to the others at a particular job. Natural language is used in requirements documents to avoid misunderstanding due to terminology differences. Individuals themselves often have difficulty articulating exactly what is “in their head”. In requirements Analysis the stakeholders often have problems describing what a system is meant to do or what they might want the system to perform. Further the users have their own expectations of what they want the system to perform; such requirements may be interpreted differently by different stakeholders such as other users, developers, project managers and analysts. Once the analyst hands over the documented and agreed upon requirements for development, they may have new interpretations associated with them, which we call a mental model of a system.

Individual users each have their own mental model. A mental model is internal to each user's brain, and different users might construct different mental models of the same user interface. Further, one of the biggest issues is the gap between designers' and users' mental models. Because designers know too much, they form wonderful mental models of their own creations, leading them to believe that each feature is easy to understand. Users' mental models of the system are likely to be somewhat more deficient, making them more likely to make mistakes and find the design much more difficult to use.

A mental model may not reflect of the system the reality it’s just the individual perception of what the system should be. This may or may not be the actual scenario. Further individual’s interactions with system may vary resulting the mental model to change along with such changes. Also mental models are specific to a system, and mental model from one system may differ to another system. Each individual will have a different view or a mental representation of a system resulting in different mental models for one system. Mental model of a system is unique for a given individual however it can have similar characteristics given that the individual will have a similar interaction with the system. The main focus of this study is to assess the mental models of the users in order to identify the how each user understands the requirements.

1.2. Motivation and Research Goals

During requirements elicitation process, the analyst to elicit and understand the needs of the user(s), It is most time consuming and difficult phase in system development. And it is the least supported. (Jeffrey & Putman, 1994; Kim & March, 1995) [31] . During this phase the analyst must understand the user expectations, and the goals of the system. (Holtzblatt & Beyer, 1995).With the proper requirements, the rest of the development process can proceed and lead to the final system. However, an incomplete requirements elicitation phase may hinder the successful completion of the rest of the development process.

Understanding customer expectations, needs are the key to success in any product development. In software we deal with intangible product and fulfilling customer needs and delivering a software product that makes the customer happy would require the analyst to identify what customer needs are. Many designers construct computer systems based on their own views of how things should be ordered for the best results. A big challenge is that the human context is generally unstructured and dynamic. Nonetheless, people tend to interact with each other, and with non-human systems, in ways that can be mapped with precision, using a simple syntactical approach. The act of making these maps is sometimes referred to as cognitive task analysis. Specifically, poor or error-prone communication between the user and analyst remains a major problem (Byrd et al., 1992; Marakas & Elam, 1998; Tan, 1994), even after several decades of research on improving requirements elicitation (Ackoff, 1967; Guinan et al., 1998). This lack for a shared understanding between stakeholders may be corrected through the use of mental models during the requirements elicitation process. Mental models have been shown to be effective at creating a shared understanding between multiple individuals in many fields (Hoover & Rabideau, 1995; Malone & Dekkers, 1984; Trochim, 1989), thereby providing an impetus to apply them to information systems development. This scenario provides us the basic framework to formulate our research goal, which is stated below.

Analyze requirements Elicitation process

For the purpose understanding stakeholders mental model

With respect to Requirements Elicitation

From the point of view of Customers

In the context of Software development industries.

The chaotic, nonlinear, and continuous nature of requirements engineering, especially conceptual modeling, warrants extensive investigations on the “people” end. Requirements engineering is characterized by ongoing sense making among stakeholders, including the sponsor of the system (managers), end-users (employees), and system analysts and. A cognitive approach that focuses on sense making processes is helpful for investigating why participants in requirements engineering understand requirements as they do [24] and why their understanding of requirements may change and shift.

2. BACKGROUND AND RELATED WORK

2.1. Mental Models

Mental models were first introduced as an internalized, mental representation of something in the world. The concept was first introduced by the Scottish psychologist Kenneth Craik (1943), who wrote that the mind constructs "small-scale models" of reality that it uses to anticipate events, to reason, and to underlie explanation. Johnson-Laird started this idea and applied it to things such as the spatial arrangement of objects (Johnson-Laird, et. al., 1998). Others, including Norman and Payne, adapted the idea for use in human-computer interaction. Mental models are psychological representations of real, hypothetical, or imaginary situations. People may misunderstand when referring to mental models, assuming mental models means, mental pictures or images, mental models have a structure that corresponds to the structure of what they represent. In many cases a mental model may contain aspects of one or more of these types of models. A user may have an image of the look of an interface, a script of the process to be followed when completing a task, knowledge of the vocabulary the system uses, and assumptions about the behavior of the system. Mental models are not mental pictures or physical models of a system, but rather the underlying knowledge structure that allows an individual to construct their perception of a system or content domain.

A mental model refers to mental representations an individual has about a system, a belief's about the system, and about the interactions an individual has with a system. A system broadly defined as "any group of items." Mental models will particularly focus on how the interactions with the system will lead to outcomes of interest. Further mental models can be stated in a much more simplified manner as; (Johnson-Laird 1983) "A mental model is an internal representation of an external reality. It is built from knowledge from prior experience, involvement with the system, or the perception that we have of the system, or how the system must react." A mental model is based on belief, not facts: that is, it's a model of what users know (or think they know) about a system such as your website. Hopefully, users' thinking is closely related to reality because they base their predictions about the system on their mental models and thus plan their future actions based on how that model predicts the appropriate course. It's a prime goal for designers to make the user interface communicate the system's basic nature well enough that users form reasonably accurate (and thus useful) mental models.

Several researchers have used mental models in an attempt to improve the requirements elicitation process. Montazemi & Conrath (1986) showed benefits of using cause-effect maps during requirements elicitation, though they did not use analysts in their experiment. Instead, the researchers created the maps after interviewing real users, and then used the maps to create an operational system for the users. While shown to be effective for understanding the user and the complex relationships that were part of the domain, as well as easy to use, the cause-effect maps were not compared to any other technique, nor were they able to incorporate non-causal relationships.

Browne et al. (1997) also used mental models to successfully elicit a higher quantity and a higher quality of information from users, and in this study there was a control group with which to compare the results. Again, however, no analyst was part of the experiment as the researchers created the maps. Massey & Wallace (1996) also found mental models to have positive effects on problem definition, this time in a group setting. Burgess et al. (1992) showed that mental models were able to assist in overcoming communication obstacles during requirements elicitation, though again, the researchers created the maps and then showed them to the users.

McKay (1998) studied real users and real analysts in an action research setting and found that cause-effect maps increased shared understanding as well as provided an overall understanding of the situation and its scope. However, there were no comparisons to other techniques or any control group, leaving the results hard to interpret. This current study will build on McKay (1998) and Burgess et al. (1992) by validating the use of mental models (specifically concept mapping) in a laboratory setting. The concept map will help the user express in a non-verbal form what is needed and help the analyst understand in a non-verbal form what needs to be done. The concept map will serve as a bridge between the user and the analyst who may come from very different backgrounds, experiences, perceptions, and styles. The issues of the mind are difficult at best to understand, but over the years a number of approaches have emerged as acceptable methods for eliciting and representing mental models. These generally fall into one of three categories as illustrated in the Figure 1.

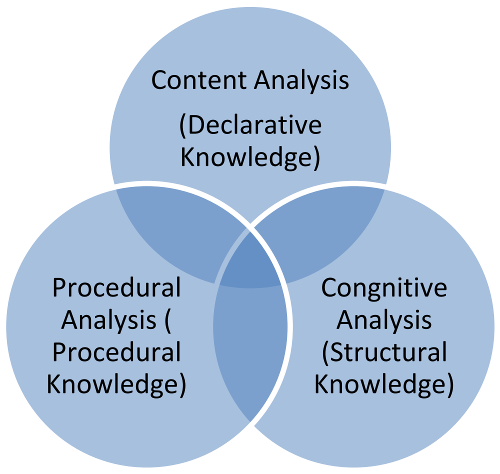

Content Analysis
(Declarative Knowledge)
Procedural Analysis (Procedural Knowledge)
Congnitive Analysis (Structural Knowledge)

Figure 1: Methods of Eliciting Mental Models

Content analysis examines written text to determine the presence and frequency of certain terms, though it typically does not provide any context of meaning or relationships between recurring concepts. Therefore it can reveal what an individual knows about a subject, but not meaningful connections within this knowledge. Content analysis is suitable for determining declarative knowledge. (Carley & Palmquist, 1992). Procedural analysis observes how an individual performs a given task, focusing on both implicit and explicit factors as to not only how, but also why certain actions are performed. This method provides good insight on the structure of what the individual is thinking throughout the process. However, this is limited to the task itself and not particularly on the general knowledge the individual may possess. It is suitable for eliciting procedural knowledge within an individual's mental model. Procedural analysis, task analysis, and similar techniques are often employed as a complement to cognitive assessment methods (Carley & Palmquist, 1992; Jonassen, 1995).

For the purpose of this study we will consider cognitive analysis, it is ideal for describing knowledge structures (structural knowledge), which includes content as well as relationships between concepts, and comparing them among a group of people. This makes this method ideal for expert/novice comparisons and classroom/training analysis (Carley & Palmquist, 1992). Researchers have long theorized about how humans mediate their internal knowledge structures and how to elicit that information. Carley and Palmquist (1992) believe that language is the key to understanding and mediating mental models. They believe that:

Both the cognitive structure and the text can be modeled using symbols, i.e. concepts. The text is a sample of what is known by the individual and hence of the contents of the individual's cognitive structure. The symbolic or verbal structure extracted from the text is a sample of the full symbolic representation of the individual’s cognitive structure. In other words, mental models can be represented as networks.

The belief that individuals interact with and represent the world through symbols has received considerable attention over the years from various perspectives (Carley & Palmquist, 1992). The notion that language is central to model formation is the basis for text-based analysis of domain knowledge, where verbal or written text descriptions from subjects is analyzed for frequency of occurrence of key terms and relations to other concepts, resulting in a network diagram showing these relationships (Carley & Palmquist, 1992).

2.2. Concept Maps

Mental models represent how well an individual organizes content in meaningful ways. Model analysis reveals inaccuracies and omissions that are crucial for deep understanding and application of course material, thus informing improvements in course design. The issue is we gather requirements and document them and verify the requirements with the customer. The main concern here is the customer may have a different vision and the analyst who gathered the information may have different views of the systems to be created. Learning theory research has revealed that individual people process and organize information in unique ways affected by their own particular experiences, cognitive abilities, beliefs, etc. for example if you take students in a classroom , will leave with twenty varying notions of what transpired during the course. The ultimate goal of the course should be to enable students to think and perform similar to experts in that field. In particular, we are most interested not merely in knowing whether they have memorized the content, but more crucially whether they are acquiring the higher-level skills required for advanced performance, problem-solving, and transfer of learning in a given domain. This requires meaningful organization of that content, making connections in ways that facilitate these skills. This can be applied to requirements analysis as the analyst as the student not knowing much of the customer's domain but contain technical domain knowledge to perform the necessary tasks.

Cognitive psychologists have developed several methods for attempting to discover how individuals "see" a particular subject. Mental models are internal conceptual and operational representations of a subject and are comprised of content knowledge (information), structural knowledge (the meaningful connections among that information), and procedural knowledge (doing useful work utilizing these connections). One effective technique is to elicit mental models and compare with that of an expert. The expert model presumably demonstrates an accurate, highly developed representation of what the subject is about. Student models are inevitably inaccurate, incomplete, and unrefined, but by discovering what they look like, instruction can be revised to address common areas of confusion and error. There are several commonly employed techniques, but the card sort has proven highly reliable, easy to complete for both administrator and participant, and provides data that can be analyzed at various levels of detail.

Some of the more common methods for eliciting structural knowledge include pair-wise ratings, card sorts, verbal protocol, and concept maps (concept mapping combines elicitation and representation in one process) (Jonassen et al., 1993; http://www.tpl.ucf.edu; Jonassen, 1995; Scielzo, Fiore, Cuevas, & Salas, 2002; Evans, Harper, &Jentsch, 2004; Subramani, Nerur, & Mahapatra, 2002). Pair-wise, card sorts, and concept maps are generally scored through quantitative methods, though there are occasional exceptions for concept mapping. Verbal protocols may also be treated similarly to quantitative written text analysis, but is often examined through qualitative procedures.

Concept mapping is a modeling procedure that simultaneously elicits and represents structural knowledge. The subject writes concepts and draws labeled links between them to indicate relationship structures. Another variation is when the researcher provides the concepts and asks the participants to generate links. As opposed to pair wise ratings and card sorts, which probe internal knowledge structures that the individual may not even be aware of, concept maps are user generated and therefore can only reveal what the participant actually recalls from memory. However, one benefit of this is that it reduces researcher intrusiveness, allowing the subject to "explicitly state the relationships they see" (Williams, 1995). Concept maps are very popular for educational applications, where teachers have students draw these diagrams to help in assessment of learning, informing further instructional needs, and providing feedback for the student (Freeman & Urbaczewski, 2002; Enger, 1998; Williams, 1995; Kinchin & Hay, 2000). One drawback to concept mapping is the learning curve involved. The "activity of concept mapping also requires instruction and practice to become "fluent" in the act of setting concepts out on paper or a computer platform" (Enger, 1998, p. 2). "Novak (1990) also noted that skill in concept mapping took at least a year to develop" (cited in Enger, 1998, p. 5). Other factors include the lack of consistency in scoring techniques and other issues (Ruiz-Primo & Shavelson, 1996; Jonassen et al., 1997).

Concept mapping is a technique for representing knowledge in graphs. Knowledge graphs are networks of concepts. Networks consist of nodes (points/Edges) and links (arcs/edges). Nodes represent concepts and links represent the relations between concepts. The concept mapping technique was developed by Prof. Joseph D. Novak at Cornell University in the 1960s. This work was based on the theories of David Ausubel, who stressed the importance of prior knowledge in being able to learn about new concepts. Novak concluded that "Meaningful learning involves the assimilation of new concepts and propositions into existing cognitive structures". [9]

Concept mapping visually illustrates the relationships between concepts and ideas. Often represented in circles or boxes, concepts are linked by words and phrases that explain the connection between the ideas, helping users to organize and structure their thoughts to further understand information and discover new relationships. Most concept maps represent a hierarchical structure, with the overall, broad concept first with connected sub-topics, more specific concepts.

“A concept map is a type of graphic organizer used to help students organize and represent knowledge of a subject. Concept maps begin with a main idea (or concept) and then branch out to show how that main idea can be broken down into specific topic” [15]

Concept mapping is useful in generating new ideas and brainstorming, in discovering new concepts and the propositions, enables to communicate ideas , thoughts and information clearly **,** integrate new concepts with older concepts, identify how the missing or inaccurate information, and gain enhanced knowledge of any topic and evaluate the information. Before we can integrate concept maps in to requirements analysis process it is important to understand how to create concept maps. The section below describes how concept maps are created and how we can integrate concept maps in to requirements engineering in order to improve understanding of customer needs.

Concept maps are graphical tools for organizing and representing knowledge. They include concepts, usually enclosed in circles or boxes of some type, and relationships between concepts indicated by a connecting line linking two concepts. Words on the line referred to as linking words or linking phrases, specify the relationship between the two concepts. The label for most concepts is a word, Propositions are statements about some object or event in the universe, either naturally occurring or constructed. Propositions contain two or more concepts connected using linking words or phrases to form a meaningful statement. Sometimes these are called semantic units, or units of meaning. Figure 2 shows an example of a concept map that describes the structure of concept maps and illustrates the above characteristics.

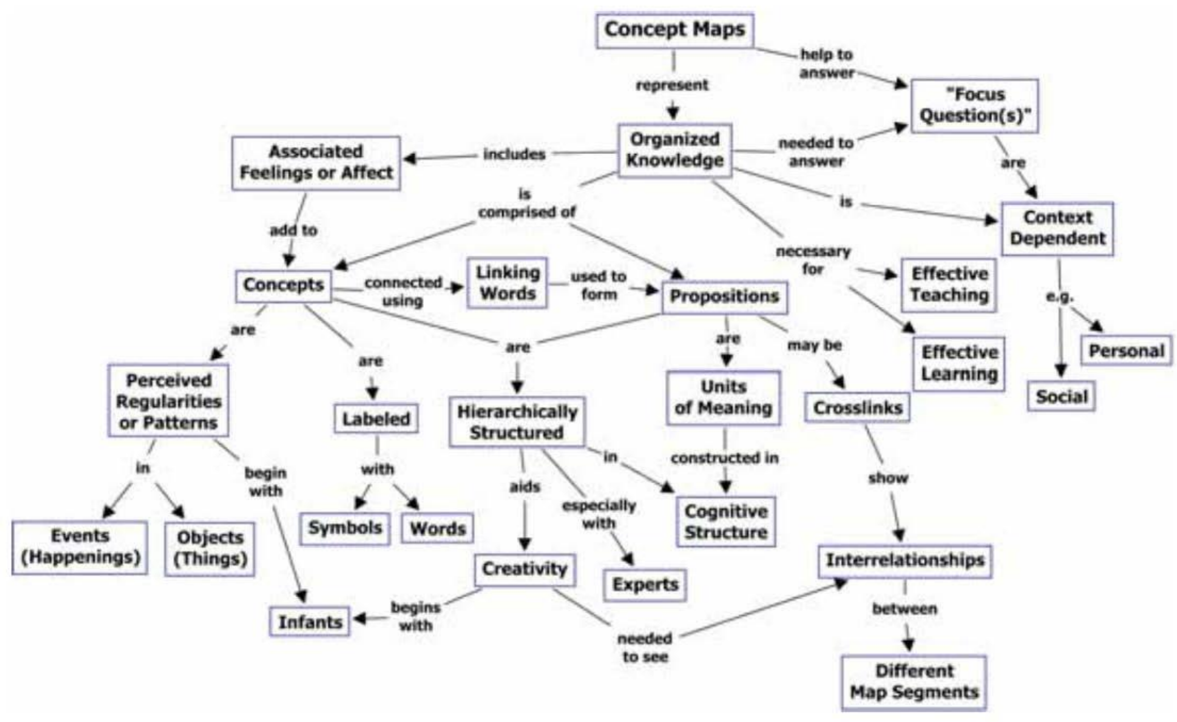


Figure 2: A Concept Map Showing the Key Features of Concept Maps. [30]

Another characteristic of concept maps is that the concepts are represented in a hierarchical fashion with the most inclusive, most general concepts at the top of the map and the more specific, less general concepts arranged hierarchically below. The hierarchical structure for a particular domain of knowledge also depends on the context in which that knowledge is being applied or considered. Therefore, it is best to construct concept maps with reference to some particular question we seek to answer, which we have called a focus question. However said that concept maps are represented in hierarchical manner, as new concepts are added or concepts are deleted, we may end with concept maps which represent either linear, network or hub type diagrams. Novak and Gowin (1984) argued that concept maps should be hierarchically structured. However, other research has shown that hierarchical structures are not always necessary (e.g., Dansereau & Holley, 1982; Ruiz-Primo & Shavelson, 1996). In this study, we also focused on the maps' graphic feature, rather than Novak's hierarchical structure. Examples of such diagrams are shown in the figure 3.

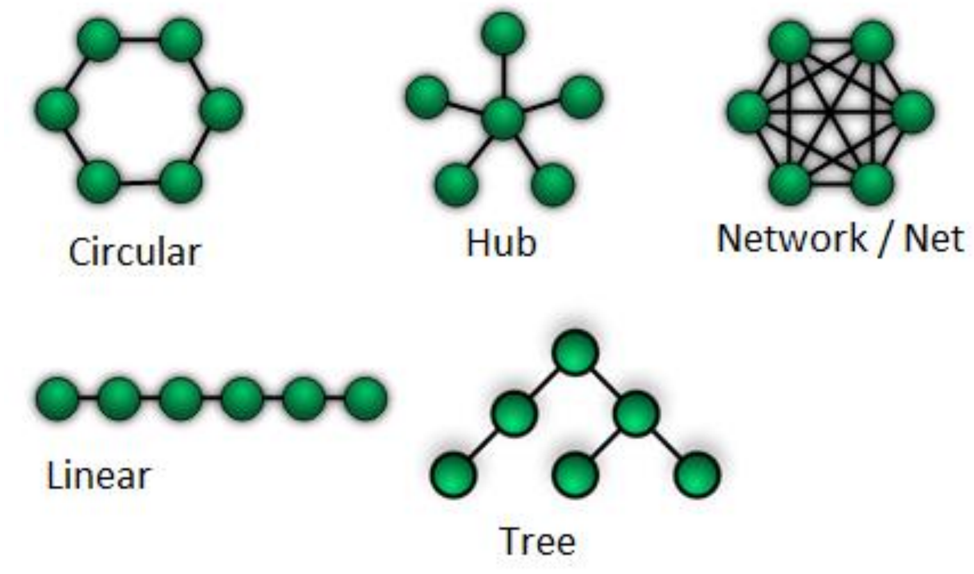


Figure 3: Concept Map Structure Complexity

Another important characteristic of concept maps is the inclusion of cross-links. These are relationships or links between concepts in different segments or domains of the concept map. Cross-links help us see how a concept in one domain of knowledge represented on the map is related to a concept in another domain shown on the map.

2.3. Measuring Concept Maps

To understand the concept maps better in requirements engineering it is important that we figure out a method to measure concept maps. Previous concept map research has focused on overall map similarity without directly accounting for the differences of vocabulary and representation that commonly occur in human-created knowledge representations. This has been reasonably effective because educational research has frequently employed closed lists of concepts and knowledge Management studies generally involve domain experts who tend to share a common vocabulary. In requirements engineering process in order to assess how each stakeholder understands a written set of requirements, matching elements is important to provide semi-automatic support for the cognitively-oriented map measures by matching the constructs in student and master maps to assess the correctness of drawn links, count levels of hierarchy, identify appropriate layers of progressive differentiation, and recognize cross-links connecting different parts of a generally hierarchical structure. Measuring concept maps poses a unique set of challengers. Given below are some of the main challenges identified when trying to compare and measure concept maps.

2.3.1. Computational Challenges

Semantic integration has received a significant amount of attention in recent years, particularly from database researchers seeking to facilitate information sharing and integrate heterogeneous data sources [21]. Most available model matching algorithms rely on assumptions. While computerized concept mapping tools employ structured representations at the implementation level, and while topics mapped by two different people will almost always be represented differently, these assumptions are not really appropriate for collections of concept maps. The “schema” of a concept mapping system consists of “concepts” and “relations” and therefore does not have any direct correlation to the entities in any particular domain. Because the schema holding a set of concept maps lacks structural clues related to the important entities in a domain, terminology variation, informality, and organizational variations are especially problematic.

2.3.2. Terminology Variation

People often use different words to represent the same concepts or the same link types. Yet neither the CMap Tools approach [22] nor Chen, Lin and Chang's (2001) matching routines [23] directly address terminology variations. The CMapTools approach sidesteps terminology variation relying on nearby terms to establish overall similarity for a map pair and the maps used by [23] were constructed with a controlled set of nodes. Because requirements are written in formal language, and written language style varies based on person to person, concept mapping systems generally do not enforce controlled vocabularies, better approaches that deal with terminology variations are needed.

2.3.3. Informality

Kremer [24] notes that there is a dichotomy between a human user's need to work with a flexible and forgiving (hence informal) system and the computer's need for a (formal) system with strong semantics. Although concept maps are intuitive and more "computationally efficient" [25] than some other forms of presentation such as pure text or predicate logic [26], Kremer asserts that concept maps can be computationally enhanced by constraining the "types" of links and nodes that can be created. Leake et al. (2002) describe concept map informality by stating that "Concept maps appear similar to semantic nets but have no fixed semantics and vocabulary" [27]. Concept maps are described by Canas et al. (2003) as a "middle point" between structural representations of CBR cases and textual descriptions. "They include structural information and are intended to concisely represent key concept properties but may not use standardized semantics. This makes them more difficult to manipulate autonomously than standardized representations but also easier to acquire when domain experts are called upon to encode knowledge" . Informality is a problematic but largely unavoidable characteristic of concept maps.

2.3.4. Organizational Variation

Constructivist learning theory asserts that there is no single correct representation of knowledge. This notion corresponds to the "cognitive shift" problem identified for concept maps scoring [23]. Two people often represent the same concepts using different but equally correct structures. However, examples of organizational variation are not provided in previous literature. Section 5. Identifies some common organizational variations we found in a human-drawn concept maps.

3. RESEARCH APPROCH

Concept maps are typically hierarchical, with the subordinate concepts stemming from the main concept or idea. This type of graphic organizer however, always allows change and new concepts to be added. It is important to recognize that a concept map is never finished. After a preliminary map is constructed, it is always necessary to revise this map. Once the preliminary map is built, cross-links should be sought. These are links between concepts in different segments or domains of knowledge on the map that help to illustrate how these domains are related to one another. Cross-links are important in order to show that the learner understands the relationships between the sub-domains in the map. It is important to note that once a hierarchical concept map can soon be changed in to a different structure altogether, for example a hub, network, and linear type of a diagram. Details of each of these types of concept map structures are presented in section 2.2.

3.1. Framework for Comparing and Calculating Concept Maps

A concept map includes nodes (terms or concepts), linking lines (usually with a unidirectional arrow from one concept to another) which are called Edges, and linking phrases which describe the relationship between nodes. Linking lines (Edges) with linking phrases are called labeled lines. Two nodes connected with a labeled line are called a proposition. Moreover, concept arrangement and linking line orientation determine the structure of the map (e.g., hierarchical or nonhierarchical) [28]. The concept maps of the requirements considered under this study is evaluated based on a scoring system. We use 4 variables in order to evaluate each concept map. Figure 4 illustrates the variable below.

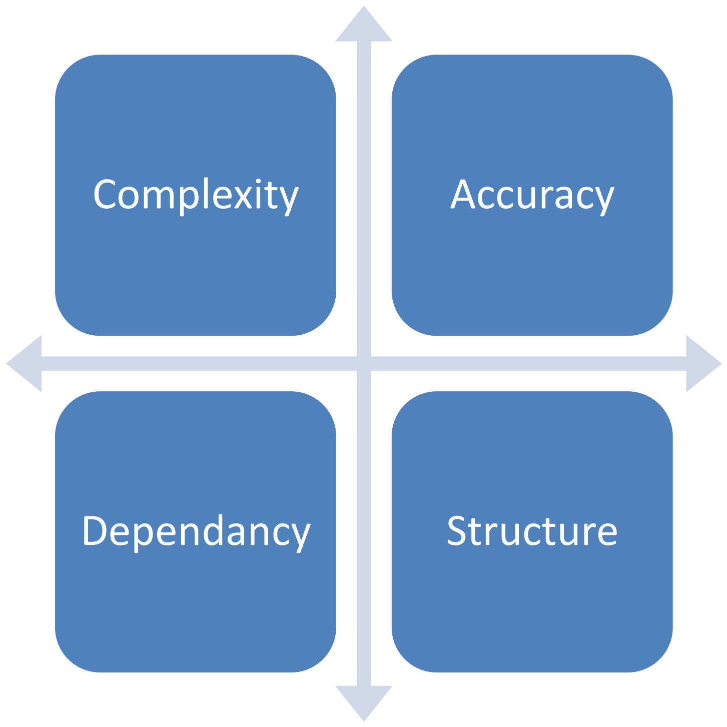


Figure 4: CADS Framework for Measuring Concept Maps

Concepts have different importance in concept maps, and the concept map layout often provides useful information for assigning concept weights. For example, a main concept usually appears at the top of each concept map, specifying the main topic We designed our assessment based on the above four criteria. Each criteria is described in Table 1 below.

Table 1: Description of CADS Evaluation Criteria

Evaluation Criteria	**Description**
Accuracy	Refers to how many nodes and Edges' are present in a given concept map.
Structure	A concept map can be of any structure, generally hierarchical (tree) , but can also be of network, linear, circular , or Hub type
Complexity	Refers to number of levels a concept map reaches,
Dependency	No of dependencies between each node. This is not necessarily refers to a hierarchical structure. Refers to the relationship between the nodes.

Based on the 4 variables listed above we can calculate the overall score of any given concept map as shown.

Let $\boldsymbol{CM} = \boldsymbol{C} + \boldsymbol{A} + \boldsymbol{D} + \boldsymbol{S}$ which represent a concept map of a given requirement problem.

Let C be denoted as the complexity of a given concept map, C can be stated as follows,

$\boldsymbol{C = no\ of\ levels\ in\ the\ concept\ map}$ Presents the hierarchical relationship between i_i and j_j. Indicates the depth of a given concept map,

Note that for circular structure concept map we assign a value of 1 for the depth.

Let A be denoted as the accuracy of a given concept map, A can be states as follows

$\boldsymbol{A = n + v}$ where *n* is the number of nodes and *v* is the no of Edges. Define the set of nodes and Edges identified by the student.

Let D be denoted as the decency between nodes in the concept map. D can be stated as follows,

D= total no of dependant nodes Define the set of nodes which are connected to each other or have a direct relationship with each other.

Let S be denoted as the structure of the concept map. S can be stated as follows

$\boldsymbol{S} = \{\boldsymbol{N}, \boldsymbol{T}, \boldsymbol{H}, \boldsymbol{L}, \boldsymbol{C}\}$Which refers to the type of the concept Map, whether the concept map falls under Network (N), Tree (T) , Hub (H) , linear (L) ,Circular (C).

As discussed in section 2.2 a concept map can be of any structure above from being a simple linear structure to a more complex network structure. This is the case when new concepts are added to the existing concept map. We identified a concept map be any one of the given structures listed above. This must be evaluated in the scoring system that is proposed in this paper. To capture the structure of the concept map we have categorized the identified structures based on its complexity, linear being the most simple structure and network being the most complex structure. The figure 5 identifies all possible structures that a concept map could take based on its complexity. Linear structure being most simple structure that a concept map could take and network structure is the most complex structure a concept map could be. When considering concept map for a particular subject different people's concept maps takes different structures. The figure 5 is sorted from simplest to the most complex structure.

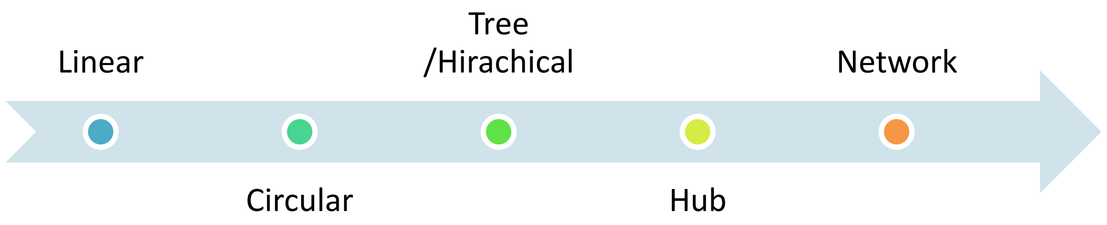


Figure 5: Concept Map Structure Analysis

The score for each structure is linear being the least and assigned the value of 1 and network being the most complex structure is assigned the value of 5. The table 2 summarizes the values associated with each concept map structure.

Table 2: Concept Map Structure Scoring Method

Concept Map Structure	**Associated Score**

Linear	1
Circular	2
Tree / Hierarchical Model	3
Hub	4
Network	5

Figure 6 is an example showing one of the detailed concept map of student concept maps (where student requirements were mapped in to a concept map) system. Based on the structure of the concept map this is classified as a network type of concept map. And the CADS valued is calculation for the below concept map is also presented.

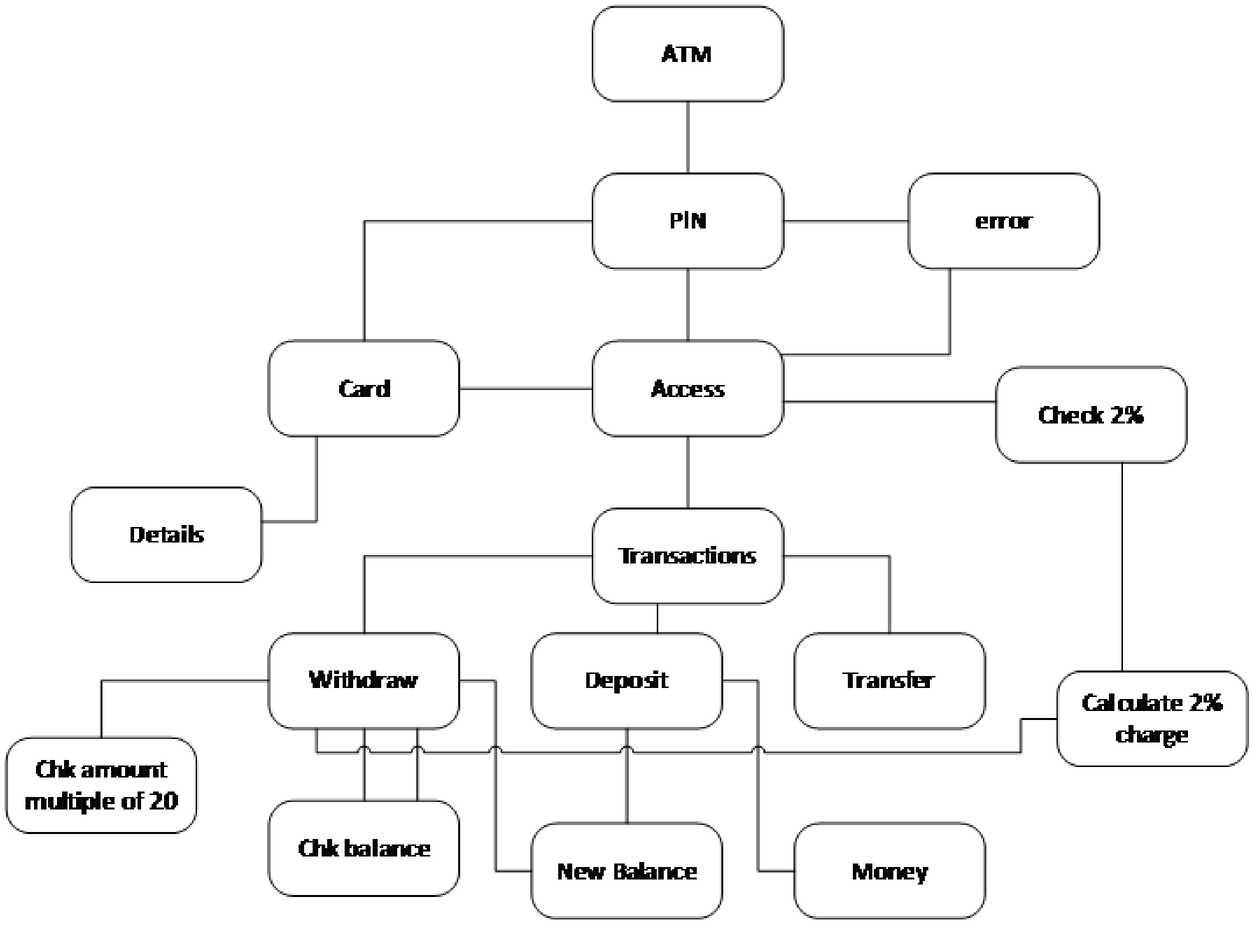


Figure 6: Student Created Concept Map

This concept map has 6 levels, The weight of the concept map is calculated as below. The first step in calculating the weight of a concept map is to access the concept map with the

CADS frame work stated in the previous section. So the overall weight is the summation of all variables of Complexity, accuracy, Dependency and Structure.

Calculating the complexity

C= 6 , as there are 6 visible levels.

Calculating the Accuracy:

A = n+v where n=16 and v=19

There are 13 nodes and 13 Edges in this concept map.

Calculating dependency:

D= 15 as there are 15 dependant nodes

There are 12 nodes which are interrelated or depended on another node.

Calculating structure:

S = T (where T denotes a Tree structure) therefore S=3

This concept map resembles a tree like structure and we have separate values considered for each type of structure a concept map can take. Therefore for this concept map we assign the value of 3 for structure.

Therefore the overall score of the concept map can be denoted as:

$$\boldsymbol{CM = C + A + D + S}$$

CM =C+A+D+S = 6+35+12+5 ➔ 58

Each of the students concept maps were scored based on the CADS framework stated above. More detail analysis is presented in the preceding sections.

4. EXPERIMENT DESIGN

The main goal of this study is to analyze the stakeholder's mental model and identify the differences between the stakeholders knowledge when gathering requirements, and to find out how effective concept maps could be utilized to make the requirements elicitation process more effective. To properly focus the research, a set research questions were needed. With the underlying goal to integrate concept maps to better understand the requirements and to identify missing or ambiguous requirements that will enable to produce software that will meet customers' expectations, the high-level questions addressed by this review was:

"How different is various stakeholder mental models of the system to be developed during requirements elicitation process, does concept map help minimize the knowledge gap during requirements elicitation process.? "

This high-level question was then decomposed into the more specific research questions and sub-questions shown in Table 1. The first research question attempts to identify the differences between the customer's mental model and the analyst mental model during requirements elicitation process. How different is our mental models when we are presented with a problem. The second research question focuses Further how well do we understand a known domain vs. an unknown domain when elicitation of requirements. The final research question will investigate how we perform using problem domain and use case domain in both known and unknown domains.

Table 3: Research Questions

Research Question	Motivation
1. Is there a difference between mental models of stakeholders during requirements elicitation process?	Investigate the significant differences in mental models of the customers and the analyst during requirements elicitation process
2. As stakeholders how do we compare in analyzing a known domain vs. unknown new domain using concept maps?	How well do we document requirements in known domains vs unknown domains using concept maps
3. What is the difference between analyzing the problem domain vs. use cases using concept maps?	The comparison between the use case method and problem domain method.

A similar experiment was conducted at NDSU [29], where the creativity process during requirements engineering process was analyzed using concept map, data from that study was relevant for this study and was used to analyze in order to compare the concept maps, details of that study is presented in this section.

A very detailed concept map and a set of requirements using use cases was constructed, the concept map was created using the requirements created by the students, this was done mainly by analyzing the student created requirements and each requirements was identified as a concept and hence the concept map for each student was constructed. It is important to note that student did not involve in creating the concept maps, and students only identified the requirements based on the problem that was presented to them. This concept map shows the requirement flow of the problems presented in appendix A.1 and A.2. This was created by a detailed discussion between a Graduate Professor at NDSU and a student with 4 years of Software industry experience. Graduate professor and student with 4 years experience are domain experts in both the system used for case study in this paper. The students were presented with the problem definition for a proposed system and were asked to identify the requirements through functional requirements and model the requirements through use cases.

Later the requirements identified through each phase were plotted to a concept map to identify each requirement. Each concept map based on student requirements and the domain expert was analyzed based on the CADS framework presented in the section 3.1,

4.1. Participating Subjects

Nine (9) Computer Science graduate students enrolled in the Software Design course at North Dakota State University in spring 2011 participated in this experiment. These students were predominantly masters and PhD computer science students and had taken requirement engineering course prior to this study. The software design course was focused on analyzing the design decisions and implementing software designs. [29]

4.2. Artifact

During this two days experiment, the participants were given a problem description on ATM machine and Communication process software and were asked to write functional requirements. Details of the study is under Appendix section (A.1 and A.2) of this paper.[29]

4.3. Experiment Procedure

Table 4: Research Method

	Familiar	**Unfamiliar**
Problem Domain (PD)	ATM (CM)	Product Verification request System (CM)
Use case (UC)	ATM (CM)	Product Verification request System (CM)

Each individual was given a two problem statements (ATM& Product Verification request System Appendix A.1 and A.2),Which reflects the known and unknown domains, where the known domain was considered as the ATM problem and the unknown domain was considered the Product Verification request System. The students were asked to write the functional requirements and model the requirements using use case diagrams in each of the problem they got. Detail diagram of the experiment procedure is illustrated in table 4. Each participant was required to list all different use cases and the requirements related to each of those use cases. Then, analyze each use case (in a rough corner) to come up with additional set of requirements. This was done to help participant generate more requirements. Further this was done taking in to consideration that requirements engineering process is an iterative process and we wanted to reflect that aspect of the process in this experiment. The correctness of the solution/use case was evaluated by a Software Engineering Graduate professor at NDSU and a Graduate student having an industry experience of 4 years in Software industry. All the participants' students knew Use Case text very well. [29]

In this experiment we will focus on analyzing the knowledge gap between the customer and the analyst. The student will be considered as the analyst and the professor from the computer science department and the industry expert with 4 years experience, who will possess domain knowledge regarding the requirements document under consideration. Each participant's requirements were placed in a detailed concept map to find the nodes of concept map covered by each participant's set of requirements. The detail map was created by a thorough discussion between a Graduate Professor at NDSU and a Graduate student with 4 years of industry experience. The CADS frame work was used to evaluate each concept map created by the students based on the requirements which were identified. And each concept map was given a score. The concepts that the students created and the concept of the expert were compared to map out the differences and identify the knowledge gap between the students (in this case Analyst) and the customer (Expert).

The scoring of requirement for each participant was divided into two Phases. In Phase I, all the requirements gathered from problem description were plotted into each node of the concept map. And a score was calculated as specified in CADS model. In Phase II, the requirements gathered from detailed Use Case text were also plotted into the concept map and the score was calculated as specified in the CADS model. Then both known and unknown domain was compared based on their score and the expert concept map created. Detail results of this experiment are presented in section 05.

5. RESULTS

All student requirements in use case and problem domain for both known and unknown domains were plotted in to concept maps and evaluated based on CADS framework presented in this paper. Given below is a summary of each students score for the concept maps created in both known and unknown domain for problem description and use case methods.

Table 5: Concept Map Scores

	Known Domain		Unknown Domain	
Student	**PD Score**	**Use Case Score**	**PD Score**	**Use Case Score**
1	68	81	93	105
2	56	68	110	127
3	47	51	61	70
4	102	119	114	125
5	86	119	116	135
6	69	94	136	155
7	80	102	93	100
8	58	101	76	96
9	100	110	86	94
10	77	84	103	117

The highlighted score was obtained by the industry domain expert and professor from NDSU computer science department. Also it is observed when you plot all requirements in to a concept maps, and there was only one student who obtained the exact score of 119 that of the domain expert, in the known domain using the use case method. Based on this data almost all students identified more requirements using use cases which resulted in more concepts and relationships when plotted in to a concept diagram. If you take a look at the highest scores on the following areas, problem description on known domain a score of 102, was obtained by the domain expert, whereas there is a tie on use case description on known domain between the domain expert and student 5, a score of 119. You can also observe that the highest score of problem description on unknown domain is obtained by the student 6, a score of 136 and the highest score of use case description on unknown domain is achieved by student 6, a score of 155. Details of the above table 5 are illustrated on charts in figure 7.

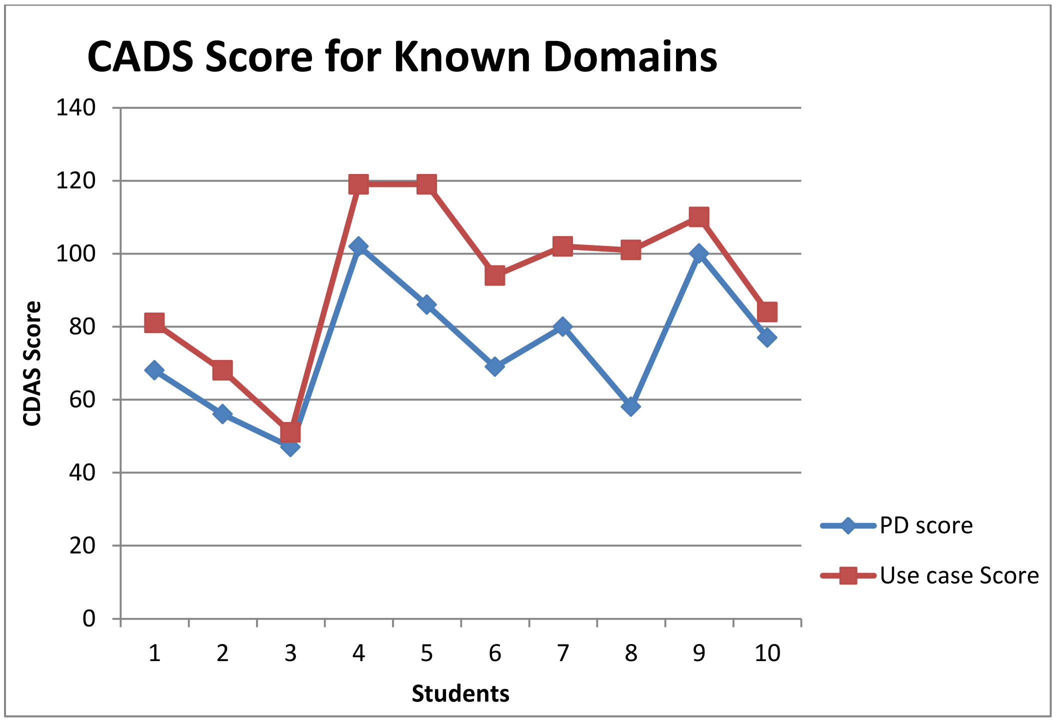


Figure 7: CADS Score for Known Domains

Based on this chart you can say both the Problem description and use case take similar pattern however all students identified more requirements using the use case method. Where use case method had yielded more concept maps.

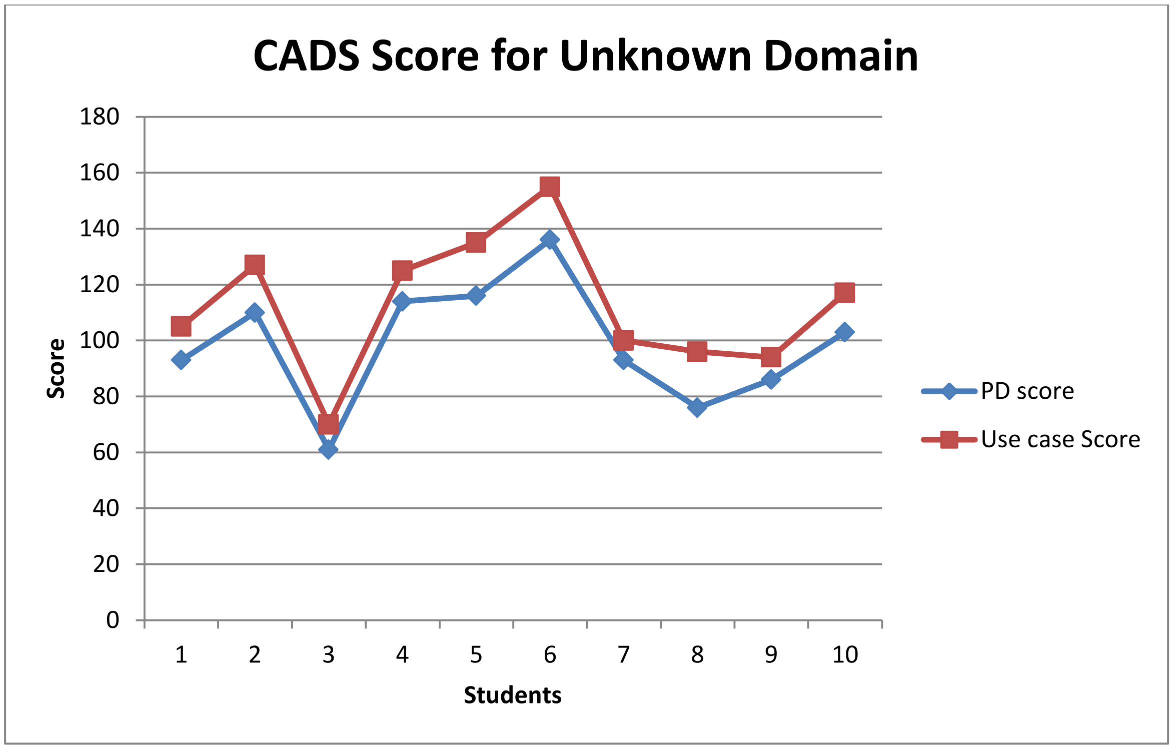


Figure 8: CADS Score for Unknown Domain

Based on the figure 8, the problem description and use case method yield very similar results, however students identified more requirements in the use case method.

Almost all of the concept maps structures plotted were similar to a network structure, with few exceptions where some of the student concept maps were more of a tree structure and a hub structure. it was also observed that in unfamiliar domain the network structure of the concept maps got more complicated resulting in more concepts and more relationship between concepts. Breakdown of each concept map scores are presented in the below table 6.

Table 6: Individual CADS Score Break Down for Known Domain

.	Known Domain							
Student	**PD Score**				**Use Case Score**			
	Complexity	**Accuracy**	**Dependency**	**Structure**	**Complexity**	**Accuracy**	**Dependency**	**Structure**
1	9	37	17	5	11	45	20	5
2	7	31	13	5	7	39	17	5
3	4	28	10	5	4	31	11	5
4	7	59	31	5	8	70	36	5
5	8	50	23	5	9	71	34	5
6	6	19	17	5	7	56	24	5
7	7	47	21	5	7	62	28	5
8	4	35	15	4	4	62	31	4
9	9	62	26	3	9	68	30	3
10	8	45	20	4	8	50	22	4

Table 7: Individual CADS Score Break Down for Unknown Domain

	Unknown Domain							
Student #	**PD Score**				**Use Case Score**			
	Complexity	**Accuracy**	**Dependency**	**Structure**	**Complexity**	**Accuracy**	**Dependency**	**Structure**
1	7	56	24	5	9	64	27	5
2	10	67	28	5	14	76	32	5
3	6	34	16	5	6	40	19	5
4	10	69	30	5	13	75	32	5
5	7	83	30	5	10	88	32	5
6	6	82	37	5	7	100	43	5
7	4	59	26	4	4	64	28	4
8	6	45	20	5	6	59	26	5
9	6	55	25	5	6	57	26	5
10	8	63	27	5	8	73	31	5

Table 8 is a breakdown of the structure of concept maps obtained in this study.

Table 8: Concept Map Structure Analysis

	Known Domain		**Unknown Domain**	
Structure	**PD**	**Use Case**	**PD**	**Use case**
Linear	0	0	0	0
Circular	0	0	0	0
Tree / Hierarchical	1	1	0	0
Hub	2	2	1	1
Network	7	7	9	9

Besides scoring concept map propositions, we also examined concept-map structure complexity. Based on the concept map structure data obtain most of the concept map diagram represent a network structure. Majority of the students in the known domain concept maps represented a complex network diagram regardless of the problem description or the use case method used. With a three (3) student's concept map represented different types of structures.

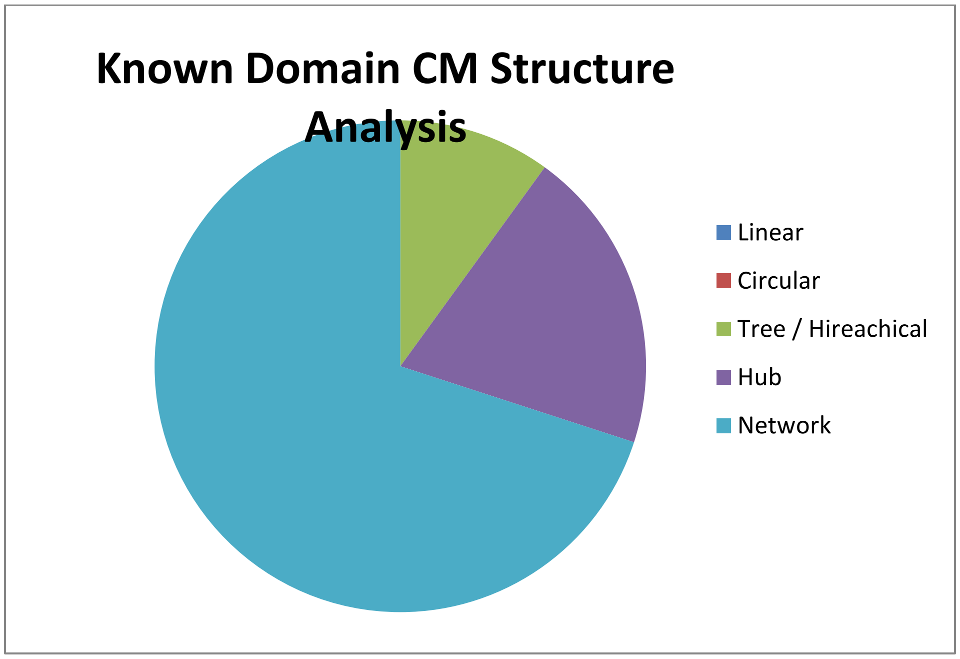


Figure 9: Known Domain CM Structure Analysis

Out of the tree students who have different concept map structure 2 were hub type structure and the other was a tree structure.

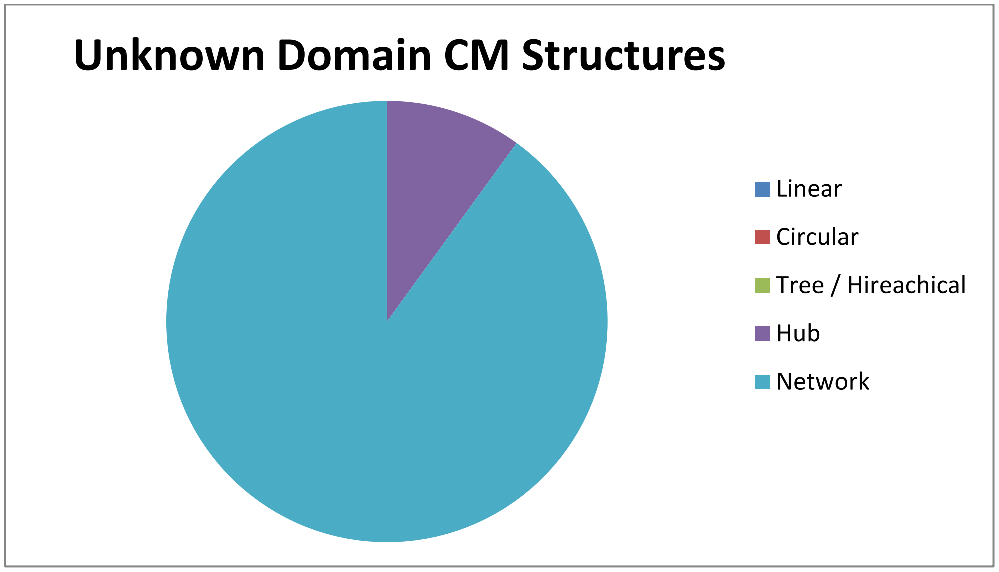


Figure 10: Unknown Domain CM Structure Analysis

When considering the unknown domain student's concept maps were network type of diagrams regardless of the methods used, problem description or use case. There was only 1 student's concept map which consisted of a hub type structure.

Table 9 presents the concept map difference between the domain experts and the students. This was calculated taking the domain expert score obtained for known and unknown domains in both problem description and use case methods and identifying the difference in scores that each individual obtained.

Table 9: Concept Map Difference

	Known Domain		**Unknown Domain**	
Student	**PD score (KPD)**	**Use case Score (Kusecase)**	**PD score (UPD)**	**Use case Score (Uuse case)**
1	34	38	21	20
2	46	51	4	-2
3	55	68	53	55

4	102	119	114	125
5	16	0	-2	-10
6	33	25	-22	-30
7	22	17	21	25
8	44	18	38	29
9	2	9	28	31
10	25	35	11	8

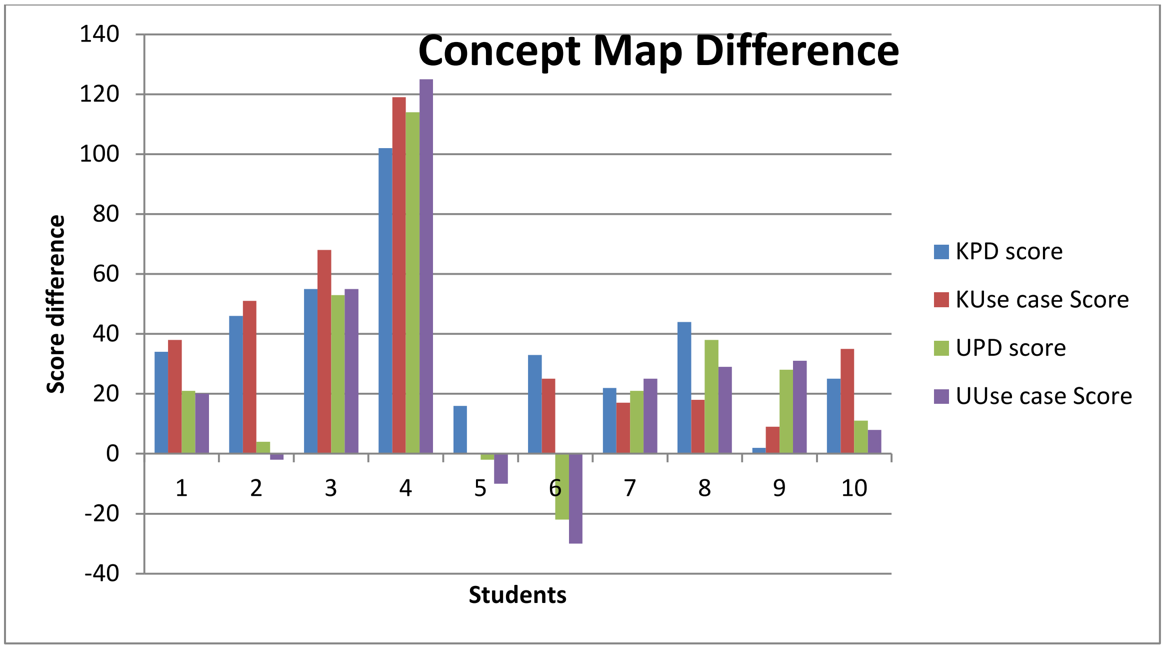


Figure 11: Concept Map Differences Analysis

This figure 11 illustrates the difference between student concept maps as compared to that of the domain expert in both areas in Known and unknown domain, using both problem description and use case methods. When you compare known domain problem description there is a significant difference between the students and the domain experts. And students have done much better in known domain using use case methods.

6. LIMITATIONS

Student misconceptions are exhibited in incorrect links and lack of organizational clarity. Removing incorrect links (for example a link identifying a BTree as a type of Binary Tree) would have increased node matching accuracy in some cases. Map clarity is also important. Identifying such ambiguous or incorrect representations would be helpful educationally and increase matching accuracy. Different people frequently use different abbreviations, synonyms, or word forms in node labels.

7. DISCUSSION AND FINDINGS

To determine whether the two concept-map assessments were equivalent, we followed our CADS framework and compared the two mapping techniques based on known and unknown domains the CADS variables of : Complexity , Accuracy, Dependency and Structure. Based on the analysis of data presented in section 5.0 we identify that student concept maps varies based on the context that was presented to them. For example students understood the more familiar domain and concept maps were more easily mapped out, it is also notable the vocabulary that the students used in known domain are very similar resulting in similar concepts, as opposed to unknown domain the concepts were not easily subtracted. There is a significant difference in scores obtained for known domain and unknown domain, also students did find more requirements in both known and unknown domains using use case diagrams.

Students performed better in known domains compared to unknown domains, however the student requirements were more complex in unknown domain as opposed to known domain. This holds true as when analyzing unknown domains student requirements are more complex which results in complex requirements statements, interdependencies and relationships, When analyzing requirements for known domains it was observes that requirements are more simple, modular, easy to understand and dependencies aware less.

We also found out that concept map structures becomes more complex as students encounter unknown domains, where the context of the system to be developed not clear, and the requirements are not that easily understood. In such scenarios student yielded more requirements in problem description method and found even more requirements using use case method.

8. CONCLUSION

Concept mapping provides a means to capture and examine human concepts, as well as a tool for aiding experts and novices at constructing a defining their own understanding of domain. When comparing concept map products, we found that concept maps can aid in requirements engineering process as a graphical tool when requirements are unclear. Further we as analysts are more comfortable in analyzing known domains for us and yield better results rather than unknown domains. When analyzing requirements for unknown domains our requirements tend to be more complex and consist of high level dependencies. This helps down to break the problem in to small manageable sections and aid in problem decomposition to understand the problem more clearly. Further the use of concept maps help decompose the problem and identify details which were not visible as a part of the bigger picture.

The CADS model described in this paper can be further enhanced to capture human nature of concept maps, taking in to consideration the language factor as discussed in the limitations in this paper. A matching system that uses structural information in addition to node and link labels is promising for matching knowledge elements in collections of concept maps. Element matching can be used to support identification of hierarchical sub-structures and cross-links in student maps.

9. REFERENCES

[1] K. E. Wiegers (2003), Software Requirements Engineering Second ed., Redmond: Microsoft Press, , p. 4.

[2] R. B. V. J. Udai K. Kudikyala (2005), "Software Requirements understanding using Pathfinder Networks as Mental Models," *Journal of Systems and Software,* vol. 74, no. 101 - 108.

[3] R. T. a. M. D. Davis (1997), "Identifying and Measuring Quality in a Software Requirement Specification," *Software Requirements Engineering IEEE Computer Society Press.*

[4] M. e. a. Keil (1998) , "A Framework for Identifying Software Project Risks," *Communications of the ACM,* vol. 41, pp. 76-83.

[5] B. a. B. V. R. Boehm (2001) , "Software Defect Reduction Top 10 List", Computer," *IEEE Computer Society Press,* vol. 34.

[6] S. R. T. a. M. D. Faulk (1997), "Software Requirements: A tutorial in Software Requirements Engineering," *Software Requirement Engineering IEEE Computer Society Press.*

[7] V. B. Hinsz (1995), "Mental Models of Groups as Social Systems Considerations of Specification and Assessment," *Small Group Research,* pp. 200-233.

[8] V. G. a. P. L. P.N. Johnson-Laird (1998), "Mental models: a gentle guide for outsiders,"

[9] D. B. K. &. Y. M. Jonassen (1993), "Structural knowledge: Techniques for conveying, assessing, and acquiring structural knowledge.

[10] J. Novak, (1991) "Clarify with concept maps: A tool for students and teachers alike. The Science Teacher,," pp. 58(7), 45-49.

[11] A. J. C. Joseph D. Novak (1991), "The Theory Underlying Concept Maps and How to Construct and Use Them".

[12] N. B. D. L. J. B. Cheng-Chih Wu (1993), "Conceptual Models and Cognitive Learning Styles in Teaching Recursion".

[13] J. B. T. a. A. Y. Nick Chater (1998), "Probabilistic models of cognition: Conceptual foundations".

[14] Models: Understanding Cognitive Change to Support Teaching and Learning of Multimedia

[15] L. A. Freeman (2002), "The effects of concept maps on requirements elicitation and system models during information systems development.".

[16] W.J. Orlikowski, D.C. Gash (1994), Technological frames: making sense of information technology in organizations, ACM Transactions on Information Systems 12 (2) 174–207

[17] D.L. Moody, G. Sindre, T. Brasethvik, A. Sølvberg (2003), Evaluating the quality of information models: empirical testing of a conceptual model quality framework, in: Proceedings of 25th International Conference on Software Engineering, Portland, OR, pp. 295–305.

[18] D.E. Avison, G. Fitzgerald (2003), Information Systems Development: Methodologies, Techniques and Tools, third ed., McGraw-Hill, New York.

[19] A.S. Huff (1990), Mapping Strategic Thought, Wiley, New York.

[20] A.R. Montazemi, D.W. Conrath (1986), The use of cognitive mapping for information requirements analysis, MIS Quarterly 10 (1) 45–56.

[21] A. Doan and A. Halevy(2005), Semantic-Integration Research in the Database Community, AI Magazine, 26(1)

[22] A. J. Canas, D. B. Leake, and Maguitman (2001) , Combining concept mapping with CBR: Experience-based support for knowledge modeling, presented at Fourteenth International Florida Artificial Intelligence Research Society Conference.

[23] S.-W. Chen, S. C. Lin, and K. E. Chang (2001), Attributed concept maps: fuzzy integration and fuzzy matching, IEEE Transactions on Systems, Man, and Cybernetics, 31(5).

[24] R. Kremer (1994) , Concept mapping: informal to formal, presented at Proceedings of the International Conference on Conceptual Structures, University of Maryland.

[25] J. G. Lambiotte, D. F. Dansereau, D. R. Cross, and S. B. Reynolds (1989), Multirelational semantic maps, Educational Psychology Review, 1

[26] J. T. Nosek and I. Roth (1990), A comparison of formal knowledge representation schemes as communication tools: predicate knowledge vs. semantic network, International Journal of Man-Machine Studies, 33

[27] D. B. Leake, A. Maguitman, and A. J. Canas (2002), Assessing conceptual similarity to support concept mapping, presented at Proceedings of FLAIRS-02, Pensacola, Florida.

[28] Yue Yin, Jim Vanides, Maria Araceli Ruiz-Primo, Carlos C. Ayala, Richard J. Shavelson (2004) , Comparison of Two Concept-Mapping Techniques: Implications for Scoring, Interpretation, and Use, presented journal of research in science teaching.

[29] Sonu Sharma , GursimranWalia (2014), Does Domain Knowledge Increase Creativity during Requirements Development: An Empirical Study.

[30] http://cmap.ihmc.us/Publications/ResearchPapers/TheoryUnderlyingConceptMaps.pdf

[31] Lee A. Freeman (2008), University of Michigan – Dearborn, USA, The effects of concept maps on requirements elicitation and system Models during information systems development

APPENDIX

A.1. Case Study 1 (ATM)

Please go through the problem description below and write a set of functional requirements for the system below. Then think (draw some use cases in rough) about the different use cases for the system and add more requirements that came to your mind while writing use case(S). While writing use case we consider alternative and exception flows and that can lead to more requirements.

Alternative flows example: Amount Exceeds Withdrawal Limit, Amount Exceeds Daily. Withdrawal Limit

The problem: ABC Bank wants to develop software for operating ATM Machine in 10 different States in United States. Customers of different banks can do a transaction using this ATM machine. ATM machine will only let one customer do a transaction at a time. However the customers of other banks (other than ABC Bank) would be charged 2% transaction fees of the transaction amount. ATM machine will have a display screen for customer interaction. It will also have a magnetic stripe so that customer can swipe his/her debit card/credit card. It will have a keypad to input the choices and the amount for the transaction. It will also have a dispenser for withdrawal of cash (in multiples of 20). There will be a printer to print the transaction details. Customers can receive the receipts for transactions done.

Customer will have to insert valid card and personal identification number (PIN) for authorization purposes. Once the customer is authorized as a valid customer then he/she can perform as many transactions as he can. ATM Machine will return card back to customer only when customer indicates that no further transactions are required. Different entities involved in the system are:

1. Customer – Any person who would interact with ATM Machine for performing transactions.
2. Bank- Bank is financial entity that validates the customer. It also validates each transaction with regards to sufficient funds to perform transaction.
3. Operator- Operator is a person who would interact with ATM Machine to start and stop ATM Machine. He/she is responsible to make sure there is sufficient cash in ATM Machine. If there are insufficient funds he reloads the machine with cash. He also removes all the deposit envelops from ATM Machine.

   a) Create a set of Functional requirements on the basis of above problem description. Please follow the below template to record the requirements.

Req. ID	**Description**
0100	A customer must be able to make a transfer of money between any two accounts linked to the card.

   b) Among the number of new requirements you Added

      i. List the requirements you created/added from your Understating of ATM machines.

      ii. List the requirements you created/added while you were writing use case(for example, while writing Alternative flow or Exception flow you realize a scenario that can happen and you realized that there is no requirements for this scenario).

A.2. Case Study 2 (Product Verification Request System - PVRS)

Please go through the problem description below and write a set of functional requirements for the system below in a similar fashion as used for the ATM system.

The Problem: The main goal of this project is to create a central repository for all communication about a product or process development so that comments, task status and work order status is available to all stakeholders involved.

The new Production Verification Request system project will assist Daktronics' design, production and process development team members in making all of the product and process development tasks performed in the design, pre-production and testing of packet assemblies transparent. It will assist the employee communication from initial design to the development process initial, through preproduction, engineering and ultimately to the stage where the assembly is fully qualified for standard production.

Current project and process communication is inconsistent. The communication that exists today is via emails, phone calls and meetings. Not all the people that need this communication are included in the emails, phone calls or meetings. Also, it is time consuming to hunt for emails, make multiple fact finding phone calls or attend multiple update meetings. Many projects are emergencies or late because of this lack of communication. If this system is developed, it will no doubt benefit entire organization.

Business Requirements: There will be six categories of users of this system: Requestors, Design Engineers, Group Coordinators, Individuals, Administrators and Guests. The users in all categories will be required to securely logon to the system.

Requestors (Project Coordinators) are allowed to enter new requests, maintain request information and status, delete requests, view the status of all requests in the system, view the request task details, add comments to any task, send emails about the requests and request tasks, and view/print/email applicable reports.

Design Engineers are allowed to enter new requests, maintain request information and status, delete requests, view the status of all requests in the system, mark tasks and needed and not needed within their group, assign responsibility to tasks and notify individuals of their responsibility, view the request task details, add new design engineering tasks to a request after it has been released, maintain their own task status information, send emails about the requests and request tasks, add comments to any task, maintain the Design Engineering task list information, and view/print/email applicable reports.

Group Coordinators are allowed to view the status of all requests in the system, mark tasks and needed and not needed within their group, assign responsibility to tasks and notify individuals of their responsibility, view the request task details, add new group specific tasks to a request after it has been released, maintain their own task status information, maintain their task group’s tasks status information, send emails about the requests and request tasks, add comments to any task, maintain their own group’s task list information, and view/print/email applicable reports. They also maintain their task group’s task status information, send emails about the requests and request tasks, add comments to any task, and view/print/email applicable reports.

Administrators are allowed to do anything that a Requestor, Design Engineer, Group Coordinator, or Individual is allowed to do. They are allowed to view the system as a Requestor would and as an Individual would. Their main view of the system should be the same as a Group Coordinator. They are also allowed to maintain users, locations, business units, groups and task groups.

Guests are allowed to view the status of all requests in the system, view the request task details, add comments to any task, and view/print/email applicable reports.

The Production Verification Request system will allow new pre-production work order requests to be entered into the system. These requests will be categorized as one of the following: a new product introduction (NPI), a manufacturing request for a mature product (MR – PCA needed for engineer testing, MN – PCA design needs to change a component to a new non-existing Daktronics part, or MD - PCA design needs to change a component to a different existing Daktronics part), a pre-production run (PPR) or a production verification (PV). This system will not be used for standard production work orders. Any request activity (newly entered, change of information, deletion of a request, Design Engineering task release or Manufacturing Engineering task release) will prompt the system to ask the user if they would like to send a message to the Group Coordinators. A new request will automatically have a status of active and will remain active until a Requestor, Design Engineer or Administrator deletes the request, or changes the status to Void or Complete. Once a request has been released to manufacturing, it cannot be deleted – only voided or completed. The request must be released to Design Engineering before it can be released to Manufacturing Engineering. When a request is released to Design Engineering, all tasks are populated to the request and the tasks appropriate to the type of request for Design Engineers are marked as needed. When a request is release to

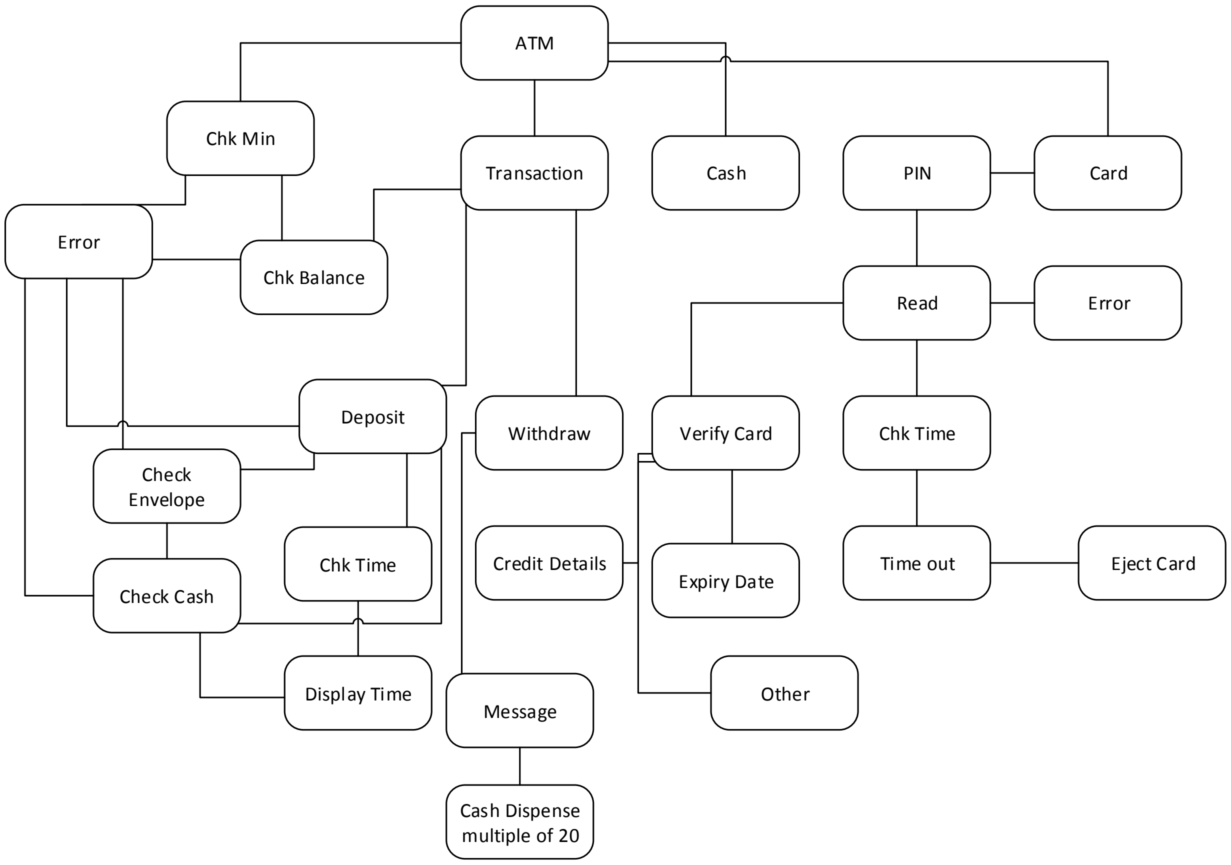
Manufacturing Engineering, the tasks appropriate to the type of request for all manufacturing groups are marked as needed.

Once a request's tasks have been released the request tasks are available to be maintained in the system.

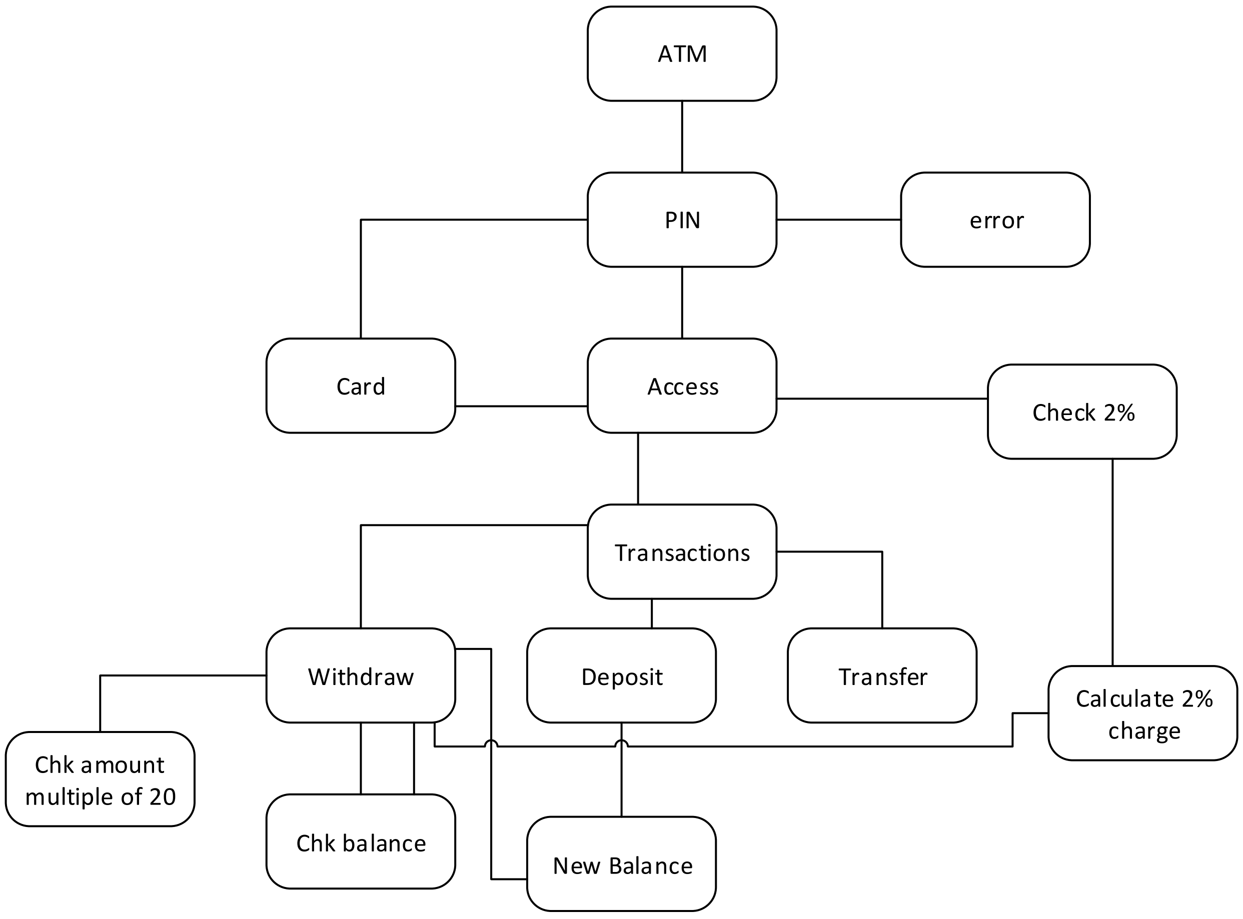
1. Create a set of Functional requirements on the basis of above problem description. Please follow the below template to record the requirements.

Req. ID	**Description**
0100	Shall implement a secure system logon

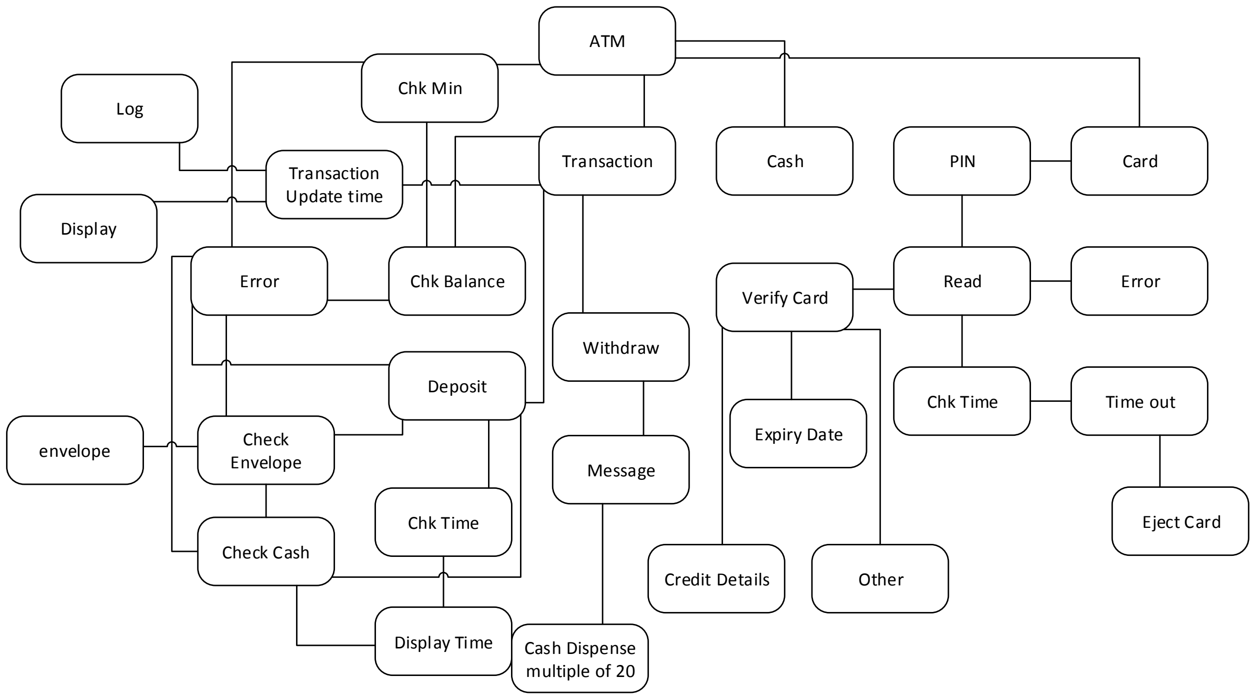
A.3. A Concept Map for a Known Domain Using Problem Description (Domain Expert)



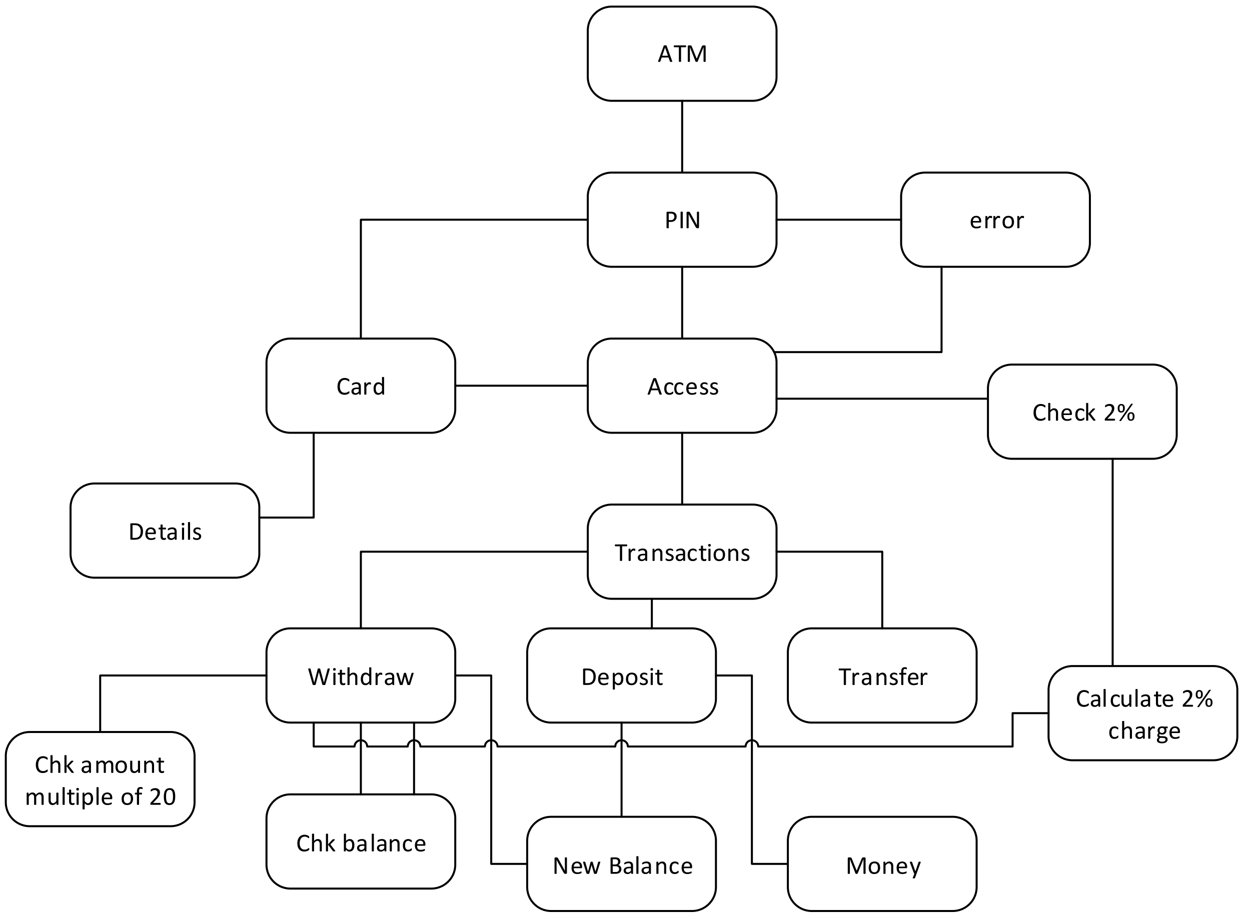
A.4. A Concept Map for a Known Domain Using Problem Description (Student)



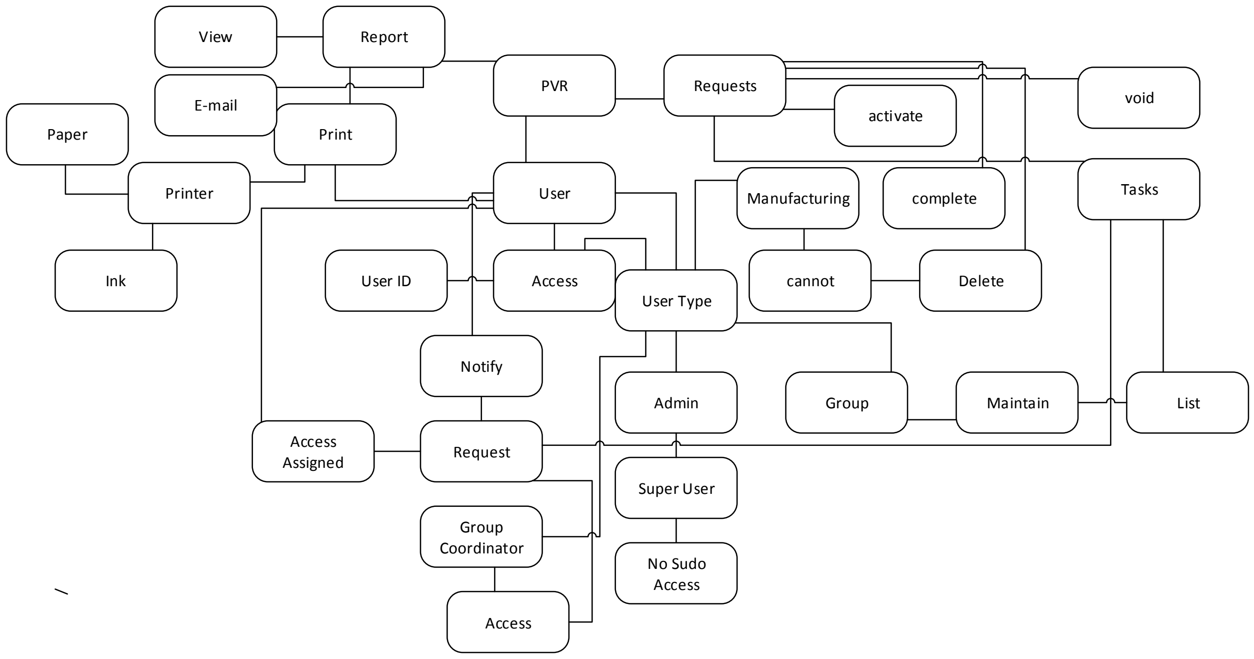
A.5. A Concept Map for a Known Domain Using Use Case (Domain Expert)



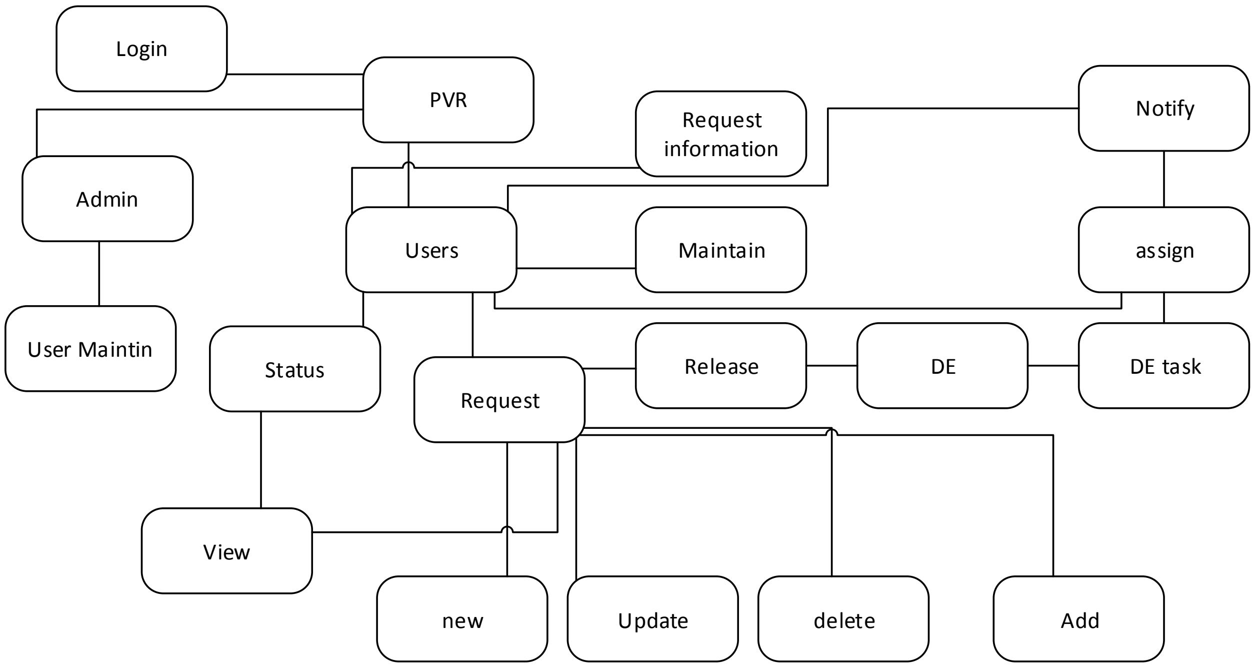
A.6. Concept Map for a Known Domain Using Use Case (Student)



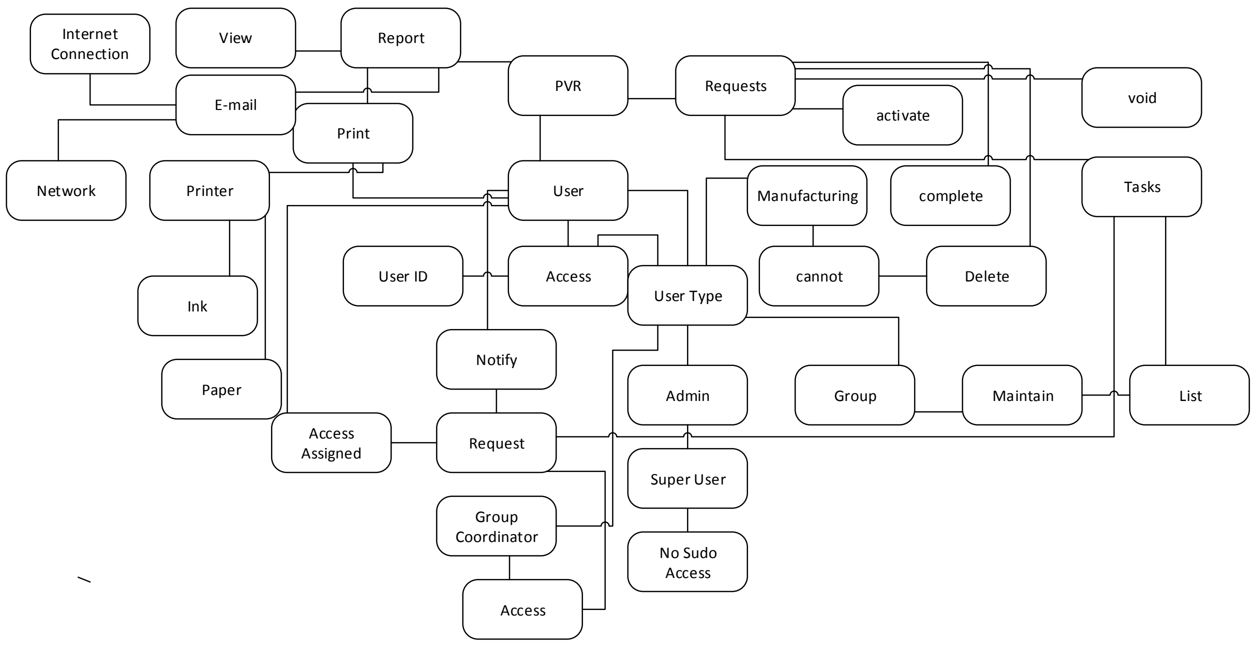
A.7. A Concept Map for a Unknown Domain Using Problem Description (Domain Expert)



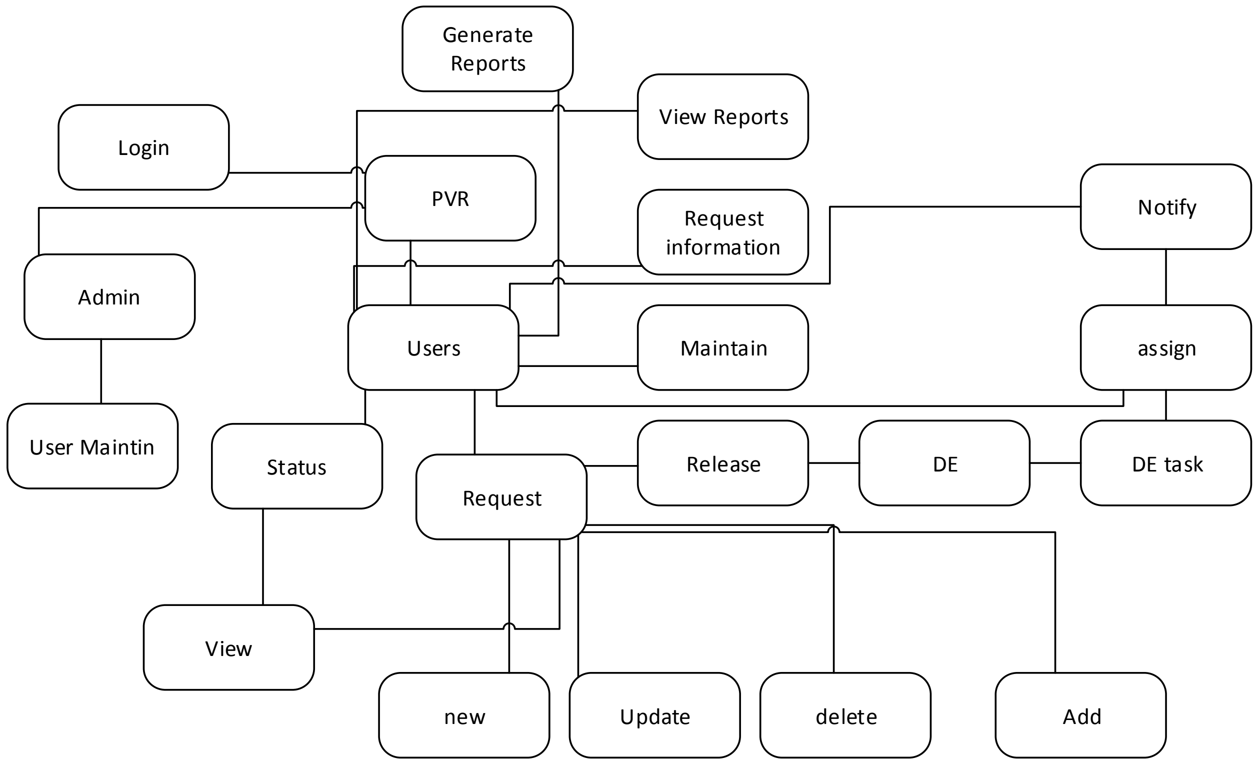
A.8. A Concept Map for a Unknown Domain Using Problem Description (Student)



A.9. Concept Map for an Unknown Domain Using Use Case (Domain Expert)



A.10. Concept Map for an Unknown Domain Using Use Case (Student)


Generate Reports
View Reports
Login
PVR
Request information
Notify
Admin
Users
Maintain
assign
User Maintin
Status
Request
Release
DE
DE task
View
new
Update
delete
Add