REDUCING FUNCTIONAL REQUIREMENT AMBIGUITIES

THROUGH A SMART ELICITATION PROCESS: A CASE STUDY

By
Silvia B. Preston
Doctoral Candidate

A Dissertation Proposal
Presented to the Faculty of the Graduate School
April 18, 2011

Dr. Ray Seyfarth, Supervisor
Dr. Dia Ali
Dr. Joe Zhang
Dr. Cliff Burgess
Dr. Tulio Sulbaran

The University of Southern Mississippi
# Table of Contents

Abstract ................................................................................................................................................. 3

Chapter I: Problem Statement .................................................................................................................. 4
  Background of the Problem ..................................................................................................................... 4
  Importance of the Problem ..................................................................................................................... 5
  Review of Literature ............................................................................................................................... 5

Chapter II: Research Objectives ........................................................................................................... 7
  Measurable Objectives ............................................................................................................................ 7
  Current Knowledge in Literature ........................................................................................................... 8
  Objectives of the Study ............................................................................................................................ 9

Chapter III: Research Approach ............................................................................................................ 14
  Research Methodology and Technique .................................................................................................. 14
  Research Timeline .................................................................................................................................. 20
  Resources Required ............................................................................................................................... 20

Chapter IV: Expected Results ................................................................................................................ 21

Chapter V: Impact of Results .................................................................................................................. 22

Chapter VI: Limitations of the Study ..................................................................................................... 23

Chapter VII: Summary ............................................................................................................................ 24

Appendix A: Case Study Protocols ....................................................................................................... 25

References Cited ....................................................................................................................................... 30
Abstract

The requirements elicitation process is a very important practice in Requirements Engineering (RE), an area in Software Engineering (SE) crucial to the development of software products. It is during the elicitation process that requirements for a software application are obtained from the customer. Customers often do not know how to express the requirements of the application to be built, causing requirements to be ambiguous. It is assumed that ambiguous requirements are one of the major causes for the development of poor software products. The development team finds it difficult to implement the requirements when what is stated is unclear. In this study, I focus on the requirements elicitation process where the primary goal is to produce good requirements with less ambiguity. Additionally, I plan to demonstrate how ambiguities can be reduced from customers’ requirements through a scripted process. This process will also aid in reducing the cognitive distance between inexperienced customers and the development team. This same process will also bridge the knowledge gap between the experienced developer and the inexperienced developer. The scripted process will be a step by step procedure to guide each stakeholder in reducing ambiguities in customer requirements during the elicitation process. In addition, the prototype of a tool will be built to support this scripted process. The tool will contain a knowledge base of previous requirements to support the elicitation process. By reducing the cognitive distance between customers and the development team, as well as between the experienced developer and the inexperienced developer, knowledge about the domain will be better understood by these two groups. The ambiguities in requirements and the lack of knowledge about the domain, between customers and the development team, provide context in this qualitative case study. The outcome of this study will illustrate how customers are able to write less ambiguous requirements that are more easily understood by the development team. In addition, the development team will be more effective in understanding customer requirements. Using the scripted process and the tool, the development team will be able to reduce any ambiguities in customer requirements if necessary. The impact of my study is relatively associated with the effort and time that goes into software development. It is assumed that less or no ambiguity in requirements means better quality in the development effort, reduced development time, and happier customers.
Chapter I: Problem Statement

Background of the Problem

In Software Engineering (SE), Requirements Engineering (RE) is one of the most important disciplines in the process of the development of software products. Successful and effective requirements engineering can improve risk management, quality, reusability, and productivity during the software development process. One of the main practices in RE is the elicitation process of software requirements. Software requirements come from the requirements elicitation process. In the elicitation process, requirements are analyzed, specified, and validated. According to the Software Engineering Body of Knowledge (SWEBOK) [1], software requirements can be defined as “a property which must be exhibited in order to solve some problem in the real world.” Requirements basically fall into two categories: 1) Functional requirements – describe the functions of the software i.e., what the software will actually do; 2) Non-functional requirements – describe the constraints of the software or the quality requirements of the software. The focus of this research is towards functional requirements. History has shown and it is a well known fact that bad requirements lead to bad products.

The first step in the elicitation process is to review the initial customer requirements. It is during the requirements elicitation process that requirements are obtained from the people involved in the process, for example, the users and customers or in general terms, the stakeholders. Often it is the intentions or perceptions of each of these players that must be properly explored to determine the exact requirements of the system. For successful software engineering, it is important for the stakeholders to have a good communication bridge. Each stakeholder has their own but very different perception of what is needed to build an effective product. Ambiguous, unclear, and imprecise requirements can be the cause of many problems in the software produced.

When customers are not able to address the requirements needed for the software to be developed, developers can become beneficial in helping customers with this process. For example, a customer may suggest searching the database for a given student name. The developer knows this may take a long time and that the student ID is required. With a suggestion from the developer, the customer agrees to the change in their requirements. What if the developer, due to the fact of being new or inexperienced, does not suggest the above scenario? Well, it would be easy to see, but the final product may work but would be very slow. A change would be necessary which causes all stakeholders problems.

The process of eliciting software requirements involves different techniques that analysts and engineer use to collect the requirements. However, these techniques might not be intuitive to customers due to their lack of technical knowledge. Requirements are often written in natural language even though notations, e.g., formal notations, diagrams, tables, patterns, and pseudo-code are available [2]. Many times the requirements produced fall short in quality, and in satisfying users’ needs. Often customers do not have the knowledge to use existing methodologies for expressing requirements. This causes poor requirements to be elicited.
Importance of the Problem

For requirements to be of quality, it is necessary that the requirements be correct, complete, precise, consistent, verifiable, modifiable, and traceable [3]. Requirements that are not of good quality often cause problems during the software development process. Interpreting requirements correctly is a major problem in RE. Studies show that only about 42%-67% of requirements are delivered in a given project [4]. Many industries cannot afford the consequences of not doing RE effectively and correctly. Ambiguous and inaccurate requirements can cost a company time, money, resources, and lost opportunities [4].

Regarding the problem of lacking of experienced personnel, there needs to be an elicitation process that can address this problem. The elicitation process must be supported by a step by step procedure that fully describes what needs to be done and by whom. The Software Engineering Institute (SEI) at Carnegie Mellon University addresses the scripted process, the step by step process for each area in the software life cycle [5,6]. This cycle covers from the requirements, design, code and test to acceptance. Although they give insights into requirement generations, they lack the details and provide no tool to help in the elicitation process.

The step by step procedure will also be essential in reducing software development efforts and costs as well as improve the quality of the requirements. This will allow the less experienced domain expert to write better requirements specifications. One approach to this is to create a reuse policy for existing requirements of existing software applications. Ambiguities of new requirements can be reduced using a knowledge base of previous requirements. Reusable requirements improve significantly the productivity and the quality of the final software product [7]. In conclusion, the elicitation process must support both a scripted process and a tool that supports machine learning. The tool must define in detail a step by step procedure and supported materials of the elicitation process. The tool must support a data dictionary and provide machine learning. How the tool is used and how to incorporate the tool's results must be described in the scripted process.

Review of Literature

Currently in the literature there is a wealth of studies that focus on the improvement of requirements specifications through a variety of methodology. In [8] the authors performed a survey on different studies that implemented methods and techniques in identifying problems in requirements. These studies provided guidelines on how to use natural language and sentence patterns processing for requirements written in natural language. The authors of this survey divided their focus into two categories. The first category describes specific language patterns for writing requirements written in natural language [9-11]. The second category characterizes the focus into linguistic rules and analytical keywords [12,13]. Although these studies offer guidelines for improving and processing requirements written in natural language, there are some restrictions that need to be taken into consideration. For example, many of these studies offer no guidance in the correction of deficiencies found in
requirements. In addition, these studies offer little to no support for reusability of existing requirements.

Other studies focus on methods for reusing requirements in different ways. In [14] the authors described software reuse to be the only practical approach that can produce the productivity increase and the quality that the software industry needs. The advantages of reusability are better when the abstraction level is raised and not only through requirement reusability, but also through designs and specifications reusability as stated in [7]. There are several approaches to requirements reusability, but as mentioned in [7], the most successful method of requirements reusability should address the three major approaches: text processing, knowledge management and process improvement. Other approaches [15,16] encompass the use of a methodology for recycling requirements by analyzing and processing existing requirements of similar systems. Another approach includes the construction of a tool for analysts to define requirements of similar systems [17].
Chapter II: Research Objectives

Measurable Objectives

This dissertation focuses on reducing ambiguities in customer’s requirements during the elicitation process of requirements. The primary goal is to produce good requirements with fewer ambiguities. In this study, an assumption is made that the current stakeholders in the process lack adequate experience. For example, the customer has never written a requirement statement. Another assumption is that the developer in the process is a less experienced developer who just graduated from college. Both have never participated in the elicitation process of requirements. Also, another assumption is that customer and developer have experienced support in each area of expertise. A script is necessary that identifies required materials and defines a step by step procedure for each stakeholder in the elicitation process. In addition, a tool will be created to support the elicitation process. This tool uses an ontology that supports reasoning-based rules and a neural network that can be trained using a reverse engineered approach from previous projects.

There are two objectives of the proposed smart elicitation process. The first goal is to identify as much ambiguity as possible from the initial customer requirements. The scripted process will describe how to use the tool and how to reduce the ambiguity. The second goal is to reduce cognitive distance between the inexperienced stakeholders lacking appropriate experiences, and their supporting experienced stakeholder. For example, when the inexperienced stakeholders lack the ability to reduce all ambiguities, the script describes the steps necessary to acquire support from experienced stakeholders like a database administrator.

In order to accomplish these objectives, the following questions must be answered:

1. How can ambiguities be reduced from customer’s requirements and converted to clearer set of functional requirements that is understood by all stakeholders?

2. What can be done to reduce the cognitive distance between the two groups: (1) the inexperienced and experienced developers and (2) the novice users and the development team when it comes to understanding functional requirements?

To resolve the problem of ambiguous requirements and to achieve cohesion between the two groups set forth above, I plan on developing a disciplined approach for reducing ambiguities in functional requirements during the requirements elicitation process. This approach will be restricted to a specific domain. The scripted process and tool I propose will focus on a database application in a university environment. The elicitation process consists of two inexperienced and two experienced stakeholders. The two experienced stakeholders, herein the non-active stakeholders are responsible for the specific project. The inexperienced stakeholders, herein the active stakeholders have never participated in an elicitation process before and both lack understanding of the process. The non-active stakeholders consult as experts in their domains: one is an experienced customer and the other is an experienced software developer with database knowledge.
The proposed tool prototype will contain a knowledge base of existing applications requirements that will be used as a requirements dictionary. The tool will support the university environment. An ontology will be used that consists of three domains: the data dictionary (also known as the SQL domain), a HTML domain, and a mapping domain. The third domain is a mapping of the relationship between the SQL domain and HTML domain. Each domain supports reasoning-based rules using Jess Rules [18]. A specific set of SQL tables are used and knowledge base instances represent each table and their associated columns. Similarly, knowledge base instances represent the HTML objects. In addition, a neural network algorithm will use the mapping knowledge base domain for training. A weighting algorithm will be developed that aids in reducing ambiguity. The neural network is invoked from the reasoning-based rules. All persistence resides in the knowledge base. Figure 1 shows the approach I plan to use.

![Figure 1 – Approach for solving the problem of ambiguities in functional requirements](image)

**Current Knowledge in Literature**

In the literature, there are studies that propose the use of ontology for the elicitation, analysis, specification and validation of requirements. The applicability of domain knowledge for requirements elicitation has been studied in [19]. In this study, requirements are elicited from requirements specifications written in natural language. In another study, [20] the authors demonstrate the use of multiple ontologies as being essential in the elicitation and reusability of requirements. These approaches have so far been restricted by complicated frameworks that have limited scopes and inability to coordinate and cooperate with other approaches.

In the area of requirement acquisition, the authors in [21] show existing work related to machine learning. Although, these works do not relate directly to my research, they depict machine learning to current software engineering approaches. In this paper, the authors show that many studies support scenario-based requirement elicitation where they use explanation-based learning methods. These studies help requirement engineers acquire and maintain the requirements with the given scenarios. Another study presented describes a tool supporting heuristic approaches that piece together partial scenarios with requirement libraries. The last study in requirement acquisition uses
inductive logic programming to extract and acquire knowledge from problem data for requirement refinement.

In [22] the authors discuss an ontology-based requirements checking tool. This tool maps initial requirements to functions in a domain ontology as input in a reasoning cycle. This cycle goes on until no new mandatory, redundant, or inconsistent requirement is found. Requirements sentences are parsed into verbs and nouns and then compared to a node in the ontology. Rules are used to reason about requirements using ontology and if there is an error, the rules determine if the requirement should be added or not added to the list. Questions are generated to customers when one of the issues is found in the requirements. Although this is a good approach, the tool requires experienced users. It is assumed the user has experience in requirements elicitation. The authors provide no further details about the possibility of reusing the ontology. Also, the reasoning about requirements is based on new requirements. There is no historical data involved in the process. Finally, the questions generated to the customer are not specific as they relate to the data.

In the area of ontology, the authors in [23], propose an approach for describing business requirements and software attributes in terms of ontologies. Ontologies are used in a semi-automated reasoning about the suitability of a certain software product. The approach proposed in this study does not provide algorithm to support the matching between the ontologies. The ontologies are built dynamically as new business requirements are specified. The authors profess that as of yet, no prior research has been done in the area of developing ontologies for existing software applications.

In another study on machine learning, in [24], the authors performed three machine learner’s tests on a reusable data set. Their goal was to improve software reusability programs by using a combination of learning techniques. The data set was tested using the following learners: association rule, decision tree induction, classification rule, and treatment learners. They concluded that the major factor for success is “Human Factors”. In addition, they found that multiple learners are necessary to identify necessary patterns in their data sets.

As per the reusability aspects concerning requirement reuse, in his study [25], Krueger produced a major study where he describes various approaches to software reuse. He evaluated the effectiveness of reuse techniques in terms of cognitive distance. He determined the most effective technique was automation of the abstractions in a reuse technique to an executable implementation. In my research, I plan to develop a semi-automated scripted process combined with a tool for reducing ambiguities in customer’s requirements during the elicitation process. In doing this, I plan to reduce the cognitive distance between all active and non-active stakeholders involved in this study. As a note, an automated process goes beyond the scope of this research. Although, with a fully trained approach, automation could be achieve.

**Objectives of the Study**

The objective of my study is twofold:

1. Create a prototype of a tool to aid in the reduction of ambiguities in initial customer’s functional requirements
Create a scripted process describing step by step how to use the tool described in part 1 in order to reduce the cognitive distance between the following two groups:

- Customers and the development team
- Experienced and inexperienced developer

Using the tool and the scripted process, ambiguities can be reduced from initial customers’ functional requirements, and the cognitive distance between the two groups can be reduced. The process to accomplish these objectives starts with a set of domain requirements, also known as the customer requirements. Given the initial domain requirements in the form of a work order (W.O.), the experts must provide successful translation into clear functional requirements. Using a portfolio of existing requirements, the tool, and the scripted process, a draft of functional requirements will be produced from the initial domain requirements as shown in figure 2. This draft supports an elicitation process providing many development options from the vague and sometimes misleading domain requirement specifications.

![Diagram of process](image)

Figure 2 – Reducing ambiguities in a new set of functional requirements

The following is an example of a functional requirement in a work order request

**Req. 1:** Students will enter their name.

From a developer’s standpoint, this requirement is very ambiguous. It is not clear what exactly is being requested. Using the tool and the scripted process, the developer transforms this requirement into a visual requirement for customer evaluation. The outcome of this transformation is as follow:

**Student Name:**

Through another iteration of the elicitation process, a preliminary set of functional requirements will be produced for development. Figure 3 shows this process in which the customer and the developer participate in the process of reducing ambiguities. The developer uses the tool and the scripted process to get the best match for the requirement in order to reduce the ambiguities.
The following is a list of choices resulting from several iterations of the process. The customer goes through this list with the developer to pick the best match for the requirement:

- First Name
- Middle Name
- Last Name
- Preferred First Name
- Preferred Last Name
- Preferred Name
- Previous First Name
- Previous Last Name
- Maiden Name
- Full Name

From this list, the customer selects

- First Name
- Middle Name
- Last Name

The preliminary set of functional requirements will be:

- **Req. 1:** Student will enter their First Name
- **Req. 2:** Student will enter their Middle Name
- **Req. 3:** Student will enter their Last Name
From this set of functional requirements, the developer can draw the requirements of how they will look on the application. Figure 4 shows the look of the requirements on the form. The way this is accomplished is through the tool. The knowledge base contains the mapping of requirements to the way they look on an application.

![Figure 4 – Requirement look and feel](image)

Finally, figure 5 shows the final steps of the process for reducing ambiguities. Through the last step of the process, upon customer approval, a final set of functional requirements will be specified.

![Figure 5 – Final process for reducing ambiguities](image)

In this final process, the final set of functional requirements will contain the details of each requirement needed for development. The description of the requirements for the example described here will result in the requirements as shown in table 1.

| Req. 1: | Input text field of length 20 for Student First Name. |
| Req. 2: | Input text field of length 20 for Student Middle Name. |
| Req. 3: | Input text field of length 20 for Student Last Name. |
| Req. 4: | A database table will be created for the application containing these fields as varchar type of size 20 each. |

![Table 1 – Detailed description of requirements](image)
With this new set of detailed requirements, the developer can proceed with the development of the requested application that will meet customer requirements. The new set of requirements will be added to the requirements dictionary for future requests.

The implemented process will be a process that learns and gains experience on specific university departments. A final requirement document will be the concluding set of functional requirements that will meet customers’ needs. The results of my research will include a disciplined elicitation process for small domains within a university, where users with lack of experience will be the primary beneficiaries. The ultimate goals of my research will be to improve functional requirements by reducing ambiguities and decrease the cognitive distance between the domain and the technical experts.
Chapter III: Research Approach

Research Methodology and Technique

In order to achieve the desired results of this study, I plan building a prototype of a tool. Table 2 shows a summary of the activities steps I plan on taking. Each step will be described in more details thereafter.

<table>
<thead>
<tr>
<th>Activity Number</th>
<th>Title</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>Activity 1</td>
<td>Identify people of interest</td>
<td>In this step, I will identify the people that will be part of my study.</td>
</tr>
<tr>
<td>Activity 2</td>
<td>Neural Network program</td>
<td>In this step, I plan creating a program using neural network algorithm to support domain mappings (see activity 5)</td>
</tr>
<tr>
<td>Activity 3</td>
<td>Ontology creation</td>
<td>In this step, I plan using Protégé [26] to create ontology to support several domains</td>
</tr>
<tr>
<td>Activity 4</td>
<td>Knowledge Base creation</td>
<td>In this step, I plan creating a knowledge base for each domain</td>
</tr>
<tr>
<td>Activity 5</td>
<td>Knowledge Base Mapping</td>
<td>In this step, I plan creating the mapping knowledge base mapping the different domains</td>
</tr>
<tr>
<td>Activity 6</td>
<td>Identify existing software</td>
<td>In this step, I will identify the 2 existing software applications that will be used to create the portfolio</td>
</tr>
<tr>
<td>Activity 7</td>
<td>Tool prototype</td>
<td>In this step, I plan creating the prototype of my tool that will incorporate the activities 2 through 6</td>
</tr>
<tr>
<td>Activity 8</td>
<td>Scripted Process</td>
<td>In this step, I plan creating a scripted process for use with the tool and the subject matter experts</td>
</tr>
</tbody>
</table>

Table 2 – Tool creation activities

Below is a detailed description of what I plan doing in each activity shown in table 2.

**Activity 1:** In this step I will identify the people of interest in my study. I will follow the proper university procedures to get authorization to work with the following people:

- 1 novice software developer
- 1 experienced software developer
- 1 expert business analyst
- 1 novice user
Activity 2: In this step I will create a neural network program to support domain mappings. These domains will be described in activity 5. The neural network program will be used to train data using previous software application requirements. Data trained will be added to a data dictionary.

Activity 3: Create ontology to support SQL domain and HTML domain using the ontology editor Protégé [26]. Jess rules will be created in this step. Jess rules allow for reasoning using knowledge supplied in declarative rules [18].

Activity 4: Create knowledge base for each domain ontology described in activity 3. Figure 6 shows the domain for the SQL tables. The HTML domain is shown in figure 7.

Figure 6 – SQL ontology and knowledge

Figure 7 – HTML ontology and knowledge base

Activity 5: Create knowledge base for the mapping between the two domains ontologies described in activity 3. Figure 8 shows the knowledge base for the domain mappings.
Activity 6: Reverse engineer pre-existing applications to create a portfolio. Requirements in the portfolio will be trained through the neural network to be mapped to new requirements.

**Requirements:**
1. Input box for student’s ID with maximum length of 11 mixed characters
2. Input box for student’s first name with maximum length of 30 mixed characters
3. Input box for student’s middle name with maximum length of 30 mixed characters
4. Input box for student’s last name with maximum length of 30 mixed characters

Activity 7: Create the tool prototype to incorporate steps 2 through 6. The tool will have a pre-populated knowledge base that will contain trained requirements from previous application.

Activity 8: Create a scripted process to be used with the tool to aid in the reduction of functional requirements ambiguities. This scripted process will guide the inexperienced developer on the process of manually reducing ambiguities from functional requirements that arrive in a Work Order format. The process will also be used to guide the developer in using the tool to further reduce ambiguities and train the requirements.

When a new application development request arrives at the desk of a software developer, the developer goes through the step by step scripted process to analyze the functional requirements. New application development requests arrive in form of a work order (WO). A WO contains the functional requirements of the new application to be built. Customers usually are the ones who write these
requirements. The following are the steps the software developer takes in order to analyze the functional requirements and reduce any ambiguities in new functional requirements:

1. Software developer uses the portfolio to manually create a hand written rough form of the initial process.
2. Developer uses the tool to enter requirements from initial requirements forms. In this step, domains will be mapped to reversed engineered software application requirements.
3. Tool generates rough requirements with ambiguities. Developers and customer review rough requirements. This step loops until desired functional requirement is produced.
4. Final step, final set of functional requirements is produced.
5. Requirements are added to portfolio.

As shown above, the scripted process and the tool will be used to reduce ambiguities in a semi-automated fashion. The tool will involve the use of an ontology and neural network combination. The process will generate a draft of functional requirements that will be used in the elicitation process and ambiguities reduction. The structure of the neural network will be one with a multilayer feed-forward network [27]. This type of network will have an input layer, a hidden layer and an output layer as shown in figure 10.

![Multilayer feedforward network](image)

**Figure 10 – Multilayer feedforward network with an input layer, one hidden layer, and one output layer [25]**

Using a two-phased backpropagation technique, data will be trained in the neural network. I will formulate mapping concepts from existing application reports to new functional requirements. These mappings define the first phased backpropagation as shown in figure 11. After this step, the developer expert will enter the customer requirements into the neural network, which will produce the functional requirement draft. With this draft, the domain experts will use a disciplined approach for elicitation to reduce any ambiguities in functional requirements. I will devise association concepts with the elicitation process to the set of functional requirements draft. Instances of these concepts support the second
phase backpropagation depicted in figure 12. With all the ambiguities reduced, the tool will produce the final set of functional requirements.

Figure 11 – First-phased backpropagation to reduce ambiguities in initial function requirements

Figure 12 – Second-phased backpropagation produces the final set of functional requirements

To understand how the neural network will act, figure 13 shows an example of new requirement specified in a WO. In this example, the requirement

**Req. 1:** Students will enter their name

will be used as the input value in the input layer of the neural network. The hidden layer of the neural network will have a list of “name” and “student name” instances. This list is the knowledge base of pre-
existing requirements mapped to SQL and HTML knowledge bases. If “student name” is not in the neural network list, but name appears in the mapping knowledge base, a neuron is created in the first layer, which fires the different elements of the neural network list. Neurons are trained and the ones containing “name” are fired. If “student name” is already in the neural network list, then neurons with “name” are fired.

Figure 13 – Example of training functional requirement and reducing ambiguities
Research Timeline

![Preliminary Timeline](image)

**Figure 14 – Proposed timeline**

**Resources Required**

In order to uncover some understanding of the questions previously outlined, this dissertation will make use of the qualitative case study research methodology. The following is a list of people that will be involved in this study:

1 novice software developer
1 experienced software developer
1 expert business analyst
1 novice user

The study will be conducted at the IT department of this university. The data to be collected will consist of two current software applications, one new set of user’s requirements, and a set of other artifacts related to the case. Data will be analyzed and the result of this study will be a scripted process and a prototype of a tool for semi automating the process of eliciting requirements and reducing ambiguities.
Chapter IV: Expected Results

The expected results of my research will include a prototype of a tool containing a knowledge base supporting inferences learned for requirement draft within a university domain. The implemented process will be a process that learns and gains experience on specific university department requirements. A final set of functional requirements document will be the concluding functional requirement specification that will meet customers’ needs. In addition to the tool prototype, a step by step scripted process will be developed to aid developers and customers in reducing ambiguities from functional requirements during the elicitation process. This scripted process will be vital to reducing the cognitive distance between the experienced and inexperienced developers, as well as between the customer and the development team. In [25], the author explains that the best way to reduce the cognitive distance is with an automated process. Given the complexity of the elicitation process and different experiences of the two groups, I will provide a semi-automated process. In theory, and by deduction, I will show the reduced cognitive distance in the elicitation process as compared to the non-automated elicitation process.

Figures 15a and 15b show a set of new functional requirements for a new application before and after ambiguities are reduced.

| Req. 1: | Application will contain Child’s name. |
| Req. 2: | Application will contain Parents’ name. |
| Req. 3: | Application will contain Employee ID. |

Figure 15a – Customer’s functional requirements before using the tool

| Req. 1: | Application will contain Child’s First Name in a input text box of 20 characters maximum |
| Req. 2: | Application will contain Child’s Middle Name in a input text box of 20 characters maximum |
| Req. 3: | Application will contain Child’s Last Name in a input text box of 20 characters maximum |

| Req. 2: | Application will contain Mother’s First Name in a input text box of 20 characters maximum |
| Req. 2: | Application will contain Mother’s Middle Name in a input text box of 20 characters maximum |
| Req. 2: | Application will contain Mother’s Last Name in a input text box of 20 characters maximum |
| Req. 3: | Application will contain Father’s First Name in a input text box of 20 characters maximum |
| Req. 3: | Application will contain Father’s Middle Name in a input text box of 20 characters maximum |
| Req. 3: | Application will contain Father’s Last Name in a input text box of 20 characters maximum |

| Req. 3: | Application will contain Faculty or Staff ID in an input text box of 11 characters maximum. |

Figure 15b – Customer’s functional requirements after using the proposed process
Chapter V: Impact of Results

The potential combination of an ontology, neural network, and a process for reducing ambiguities in requirements during the elicitation process, imply the production of a better set of functional requirements. Many studies have shown that a good set of functional requirements produces a good software product [4,31-33]. Given the semi-automated process, we can assume a shorter requirements elicitation process. In addition, with the improved quality of the functional requirements, one can assume better quality in the development effort and a reduction in the development time.

Using the scripted process in combination with the semi-automated tool during the elicitation process will benefit the development team and the users who write the functional requirements. Users will learn how to write better requirements that will have less or no ambiguity. The process of learning how to elicit functional requirements using a scripted process will come from the meetings between users and developer experts. The process for reducing ambiguities in requirements during the elicitation process using computational learning with ontology will be the intellectual merit of my research. The broader impacts of my study will be the coordinating of outputs with improving requirement quality through an ontology.

In theory, and by deduction, I will show that with the use of a scripted process and a tool, functional requirement ambiguities can be reduced in a semi-automated smart elicitation process. In addition, with the use of a scripted process and the tool, the cognitive distance can be reduced between two groups: (1) customer and the development team and (2) the experienced and inexperienced developer.
Chapter VI: Limitations of the Study

The research described here is just a demonstration of my theory. The complete implementation of my theory in a university domain would be too big to put into practice. This study is limited to focusing on misused and misunderstood parameters between domain and engineering experts from various departments. For example, if you are graduating from a PhD program, you must have an application for graduation and an application for the graduation commencement. The extension of my study includes coordination between applications that are related. In the future, this study could be extended to the complete software design process. Also, the theory here could be used to map requirements to software design during requirements elicitation.
Chapter VII: Summary

Chapter I provided the background to the problems concerning functional requirements and the importance of clear and concise requirements in software development. Additionally, I described the problem that drives the study of this dissertation and what has been found in the literature regarding to the problem. In Chapter II the objective of this study was described in more detail along with the literature content related to my research. A brief statement of my research objective along with the research questions was also provided. In this same chapter, I presented the measurable objectives of my study and how my proposed work will perform. The research methodology and technique of my study were given in Chapter III. Also in Chapter III, a proposed timeline to complete my study and the resources required to complete this study were given. The expected results of my study were provided in Chapter IV. In Chapter V, I described the impact of the results of my study in the field of study. Finally, possible extensions and limitations of my study were presented in Chapter VI.
Appendix A: Case Study Protocols

The following applications will be used in this case study. The requirements in these applications will be used to build a knowledge base and used in the initial training of the neural network.

Figure A1 – Contact Graduate Reader [34]

Table 1 and 2 list the functional requirements for the Contact Graduate Reader application shown in figure A1.
Work Order Request # 169030

I need a form created for our website for students to submit to our graduate reader. Please use the information below.

Candidates for a doctoral degree, please complete this form and check for accuracy as this will be used to print the commencement program for graduation.

Full Name:____________________________
City/State:____________________________
Degree to be received:_______________(drop down menu)
Doctor of Education
Doctor of Musical Arts
Doctor of Philosophy

Last degree received:_______________
from:__________________year:___________________
Dissertation Title:_______________________________
Major Professor’s Name:_________________________
Department:__________________________________

Submit Button- sends it to xyz@usm.edu

Table 1 – Original request

Work Order Request # 190835

We need to update the Graduate Studies Doctoral Degree Form. Under Degree to be received, the next tab needs to be: Date of Graduation...with a drop down of Month and next drop down of Year. Also please add an asterisk by middle name. and collect student e-mail address.

Table 2 – Follow up requesting changes
Tables 3 and 4 lists the functional requirements for the Music Audition Application shown in figure A2.

Work Order Request # 207199

Needs a form created for the school of music website. This is a form for prospective students to fill out for audition days. This will be a basic form, and send out a message after it has been submitted. Would like attachments to be put on an e-mail response to the student after the form as submitted. The first audition day is Nov. 13.

Hi Silvia,
Below is the form we would like:

The University of Southern Mississippi School of Music
2009-10 Audition Dates

First name:_________________________
Last name:_________________________
E-mail:____________________________
Phone:____________________________
Address:___________________________
City:______________________________
State:________Zip:__________________
Year of graduation:_________________
School name:_______________________
Voice part or instrument:______________

SAT/ACT score:____________________
Intended major:_____________________

Ensembles you have participated in:

__________________________________

Audition date I plan on attending:
______NOVEMBER 13_______ JANUARY 29_______MARCH 5

_____(checkbox) I would like to receive a hard copy of packet. (An electronic packet will be automatically e-mailed to you).

I’d like for all the blanks to be required. Once students hit “submit” they would receive a confirmation email saying that their form has been received. If possible, I would also like to include some attachments (welcome letter, schedule, campus map, etc) in the auto reply to the prospective student. The submitted form would go to “music@usm.edu”

I would like for the form to fit into our School of Music Web site template. I don’t want it to look like they are going to another site.

Please let me know if you have any questions.

Thanks,

Table 3 – Original request

Work Order Request # 234378
I need to make a few minor changes to the form below:
https://www4.usm.edu/music/musicauditionapp/.
1. Take off "March 5, 2010" text.
2. Add "2010-11" infront of School of Music Audition Application text at top and make text larger.
3. Make the following the first field with the option to select one date only.

Audition Date:
- Nov. 12
- Jan. 28
- Feb. 25
- March 25

I'd also like to have 4 attachments sent to those who register. Will send those once contacted by you guys.

This is a hot job and needs to be completed by July 22.

Thanks!

Table 4 – Changes request
References Cited


[34] https://www4.usm.edu/graduateschool/contactgr/, “Contact Graduate Reader,” University of Southern Mississippi, 2010.