Empirical Evaluation of a Visualization Tool for Knowledge Engineering

by

Mary Margaret Allen
B.Sc., University of Victoria, 2000

A Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

in the Department of Computer Science

We accept this thesis as conforming
to the required standard

[redacted]

Dr. M.-A.D. Storey, Supervisor (Department of Computer Science)

[redacted]

Dr. D. Olesky, Departmental Member (Department of Computer Science)

[redacted]

Dr. D.H. Damian, Departmental Member (Department of Computer Science)

[redacted]

Dr. J. Gennari, External Examiner (Department of Medical Education and Biomedical Informatics, University of Washington School of Medicine)

Supervisor: Dr. Margaret-Anne Storey

ABSTRACT

The lack of methods and models for evaluation has long been a challenge in software development. Without empirical investigations into real practice, the potential of many new technologies may remain unexplored. While tools that support the knowledge engineering process through information visualization are being created, there is an absence of established evaluation theory or task models in this domain. Through contextual observation and an introspective case study, this thesis demonstrates the feasibility of using requirement elicitation techniques to guide the refinement and evaluation of such a tool.

Through the analysis of our qualitative results, we derive a descriptive taxonomy of tasks performed by users of frame-based knowledge representation systems. Suggestions for cognitive support for some of these tasks, including potential visualization support, are also proposed. We further provide some more experimental evaluation designs based on these results, which can be used to investigate the theories we propose. The Shrimpbib knowledge management tool built as part of a case study is also an important contribution. It uses visualization techniques to share knowledge about academic references amoung members of a small research group.

Examiners:

[redacted]

Dr. M.-A.D. Storey, Supervisor (Department of Computer Science)

[redacted]

Dr. D. Olesky, Departmental Member (Department of Computer Science)

[redacted]

Dr. D.H. Damian, Departmental Member (Department of Computer Science)

[redacted]

Dr. J. Gennari, External Examiner (Department of Medical Education and Biomedical Informatics, University of Washington School of Medicine)

Table of Contents

List of Figures

Acknowledgements

I would first and formost like to thank my supervisor, Dr. Margaret-Anne Storey, for all her encouragement, guidance and advice throughout the last two years, and especially for her help preparing this thesis. She makes many impossible things possible, and possible things, a pleasure.

I would also like to extend special thanks to Dr. Cornelius Rosse and all the members of the Structual Informatics Group at the University of Washington for taking part in this research, and for their hospitality during my visit to their group. Without their thoughtful participation and feedback this work would not have been possible.

Thanks also to Dr. Daniela Damian for her guidance and advice, especially regarding the portions of this thesis related to requirements engineering.

The members of the CHISEL group have all added immesuarbly to my experience as a graduate student, both personally and academically. Thanks go to all of them, and especially to Elizabeth Hargreaves and Jeff Michaud for feedback on this thesis, and to Neil Ernst for his enthusiasm and work on the Shrimpbib project. Finally, thanks to all family members and friends whose moral support, friendship, and proof-reading skills made this work possible and enjoyable.

Chapter 1: Introduction

The lack of methods and models for evaluation has long been a challenge in software development [1]. While other fields of science and engineering have well-established research paradigms, the field of software engineering has not yet developed this sort of guidance for researchers[2]. Without these methods and models, the field has become rife with technology-driven projects that fail to address fundamental issues, systems designed by introspection, and good ideas neglected after faulty implementations. Such technologies can be described as 'toys': while they may be attractive and exciting, seldom is their use principled and applied for a specific purpose and, in many cases, their real potential remains unexplored. These technologies can only make the transition from interesting ideas to useful tools through careful empirical investigations into actual practice [3].

The Jambalaya tool, developed at the Computer-Human Interaction and Software Engineering Lab (CHISEL) at the University of Victoria, is an example of a technology making this transition. This thesis details efforts to refine Jambalaya from a toy technology into a useful knowledge engineering tool by looking at how to evaluate information visualization tools for knowledge engineering. In particular, it demonstrates the feasibility of using techniques borrowed from requirements engineering to guide the exploratory stages of the evaluation of such tools.

The field of knowledge engineering examines systematic approaches to constructing knowledge-based systems. The knowledge bases these systems employ must support computer inference algorithms to allow them to replicate human behaviour. However, they must also be understandable to humans, so that knowledge can be modeled and later reused. In practice, they are frequently too large and complex for people to understand easily.

Information visualization has been shown to aid human cognitive processes in many domains. Jambalaya uses interactive visualization techniques aimed at helping knowledge engineers better understand the complex knowledge bases they work with. Its interactive visualization features were adapted from a visualization tool for program comprehension called SHriMP [4], with the belief that features that support the

navigation of hierarchical code bases would also support navigating hierarchical knowledge bases. The tool was therefore designed with at least a high-level purpose in mind; this places it further along a continuum of toy-to-tool transformation than many technologies. However, as is the case in many software projects and particularly in academic projects, no investigation into the working practices of knowledge engineers was conducted prior to Jambalaya's implementation.

To assess the usefulness of the tool and to guide its future development, empirical evaluation was therefore required. However, a review of evaluations conducted to date in both the fields of knowledge engineering and information visualization revealed that neither of these young fields provides established methods for this kind of investigation. Our own early attempts to evaluate Jambalaya by verifying hypotheses about the tool revealed that too little theory has been developed in this area to build relevant hypotheses. However, evaluation can still guide the evolution of our tool. As pointed out by Almstrum *et al.* [3],

> "Evaluation must often provide data in the absence of a hypothesis, as one step in a process that leads to hypothesis-building and ultimately theory-building."

Such exploratory evaluations use ***data-driven analysis***, in which data are examined for emergent patterns, instead of ***theory-driven analysis***, where hypotheses derived from theory are tested.

The approach taken in this thesis uses data-driven analysis to determine the tasks involved in building knowledge bases, and how knowledge engineers might benefit from cognitive tool support for accomplishing those tasks. It borrows techniques from the field of requirements engineering, including an empirical observation case study of expert knowledge engineers and an introspective case study conducted within the CHISEL group.

We synthesize and summarize the results of these two case studies to show how the use of requirement techniques has helped to build theories and hypotheses that can be verified using more theory-driven approaches. This synthesis provides a preliminary taxonomy of tasks performed by the developers of knowledge-based systems. This taxonomy has implications both for visualization tools in this domain (including Jambalaya) and for future empirical evaluations in this area.

The contributions from this thesis include:

- a descriptive taxonomy of tasks performed by users of frame-based knowledge engineering systems;
- suggestions for cognitive support for some of these tasks, including visualization support;
- proposed experimental designs for future evaluations of information visualization tools for knowledge engineering, including hypotheses about the cognitive support Jambalaya offers; and
- the Shrimpbib tool created during the introspective case study in Chapter 6;

These contributions are important as more and more visualization tools are being developed for the field of knowledge engineering with very little assessment of their real potential use and value. Members of the CHISEL group are continuing to refine the Jambalaya tool based on these findings to make the tool more useful to knowledge engineers, and are also continuing investigations that we hope will help to establish the requirements for visualization tools in this domain.

1.1 Thesis Outline

The second chapter of this thesis provides some background and definitions of key concepts in the field of knowledge engineering, with a discussion of the challenges to empirical evaluation particular to this field and a review of empirical investigations in this domain. Chapter 3 introduces the field of information visualization, including the evaluation techniques used in that field. It also examines visualization tools created for the knowledge engineering domain, including a detailed description of Jambalaya, and the empirical evaluations that have been carried out on these tools to date.

Chapter 4 is a pivotal chapter; in it we step back to look at the previous evaluations in both fields in the context of possible evaluation approaches and techniques. We then look forward at the data-driven approach taken in this thesis. The approach is described with respect to the field of requirement engineering, from which our techniques are borrowed.

Chapter 5 details how empirical observations of expert users revealed insights into the tasks involved in knowledge engineering. Contextual observation techniques were used to investigate knowledge modeling activities conducted by the University of Washington Structural Informatics Group working on the Foundational Model of Anatomy project. These observations yielded detailed task analyses that can guide development of knowledge engineering tools, and provided insights into the problems faced by knowledge modelers and the end users of the knowledge bases created. The group further provided feedback on Jambalaya and other possible visualization techniques that might support their work.

First-hand experience adds invaluable insight into the issues and problems in any domain. An introspective case study that was conducted to gain such insight into the requirements for cognitive support tools in the knowledge engineering domain is described in Chapter 6. In the case study, knowledge elicited from the CHISEL group concerning the academic literature read and shared by the group members was stored in a knowledge base. The case study explores how Jambalaya can be used to examine the entities and relationships in the knowledge base, to answer practical questions and add value to a knowledge-based system created for the group. While this case study served to evaluate the use of Jambalaya in the context of a real project, the Shrimpbib tool that was created is also an important contribution of this thesis, and continues to be refined and used by the CHISEL members.

In Chapter 7 we synthesize our results to provide a preliminary taxonomy of tasks performed by the developers of knowledge-based systems. The implications of this taxonomy for Jambalaya, for other visualization tool support and for future empirical evaluations in this area are discussed. This section also proposes some potential experimental designs for information visualization tools supporting the knowledge engineering process.

The thesis concludes in Chapter 8 with a summary of this thesis' contributions, and areas of ongoing and future work.

Chapter 2: Knowledge Engineering and Knowledge-Based Systems

In this chapter, some key concepts from the domain and a brief history of the knowledge engineering field are presented as background information for the rest of this thesis. This chapter examines the challenges to empirical evaluation research in this field and a review of techniques that have been used to overcome them. This discussion is important as the evaluation of Jambalaya may face many of the same challenges.

2.1 Key Concepts and Definitions

There are many approaches and methodologies for creating knowledge-based systems. The ***Protégé-2000*** tool, with which Jambalaya is integrated, uses a ***frame-based representation system***. In these systems, ***ontologies*** are used to explicitly model conceptual domain knowledge, and computer inferencing operations are carried out using this domain knowledge and libraries of re-usable problem-solving methods.

2.1.1 Ontology

Within the knowledge engineering community, many different definitions of the term 'ontology' have been put forward [5]. A general consensus seems to exist that an ontology formally describes the concepts in a domain, and the relationships between them, in a way that is both expressive enough to be understood by humans and precise enough to be used by computers. According to Studer [6],"...ontologies are put forward as a means to share knowledge bases" between knowledge-based systems. Without an ontology, terms and symbols in a domain may be ill defined, ambiguous or confusing.

According to Noy and McGuiness, some of the reasons for creating ontologies are [7]:

- To share common understanding of the structure of information among people or software agents.
- To enable reuse of domain knowledge.
- To make domain assumptions explicit, so that such assumptions are not "lost" in a programming language as a system matures.

- To support automated analysis of domain knowledge.
- To aid in human decision-making.
- To mediate communication between knowledge and domain engineers.

2.1.2 Protégé-2000

The Protégé environment [8] has been developed at Stanford University over the past 16 years, and is being actively used by hundreds of users worldwide in many knowledge domains. It is therefore a rare example of a mature and well-adopted knowledge engineering tool. Protégé is a general-purpose tool that allows analysts to build knowledge-based systems by creating and modifying reusable ontologies and problem-solving methods, and by instantiating ontologies to construct knowledge bases. It supports the modeling of ontologies and the use of ontologies to guide acquisition of content knowledge from subject-matter experts.

2.1.3 Frame-based systems

The following discussion of frame-based systems, and how they are used to represent knowledge bases, is adopted from "Ontology Development 101" by Noy and McGuiness [7]. It uses screenshots of the Protégé-2000 system to illustrate how that system works, and a fairly accessible knowledge base that describes the domain of wines, wineries, regions and food accompaniments for wines. This knowledge base was selected for this example because of its relative familiarity to a non-expert in knowledge engineering, and because it is a standard example used in many Protégé papers, including the original work from which this discussion was adapted. It will be used as a running example throughout this thesis.

Protégé supports a representation system based on frames, which are data structures that describe the properties and relations of an entity [9]. A knowledge base in a frame-based system has two parts: an ontology defining both the hierarchy of type definitions and the relationships between the types, and a collection of instances, or instantiations of those types. To build such a knowledge base, one possible methodology involves deciding the domain and scope being modeled and examining existing ontologies for

possible reuse before enumerating the important terms in a domain. These are the terms that will have statements made about them or that will be explained to the user.

Each of these concepts will be modeled as a *class* in the class or type hierarchy. In the wines example, the classes "white wine", "rosé wine" and "red wine" are all subclasses of the "wine" class, since they are all types of wine. *Subclasses* inherit properties from *superclasses* to form this hierarchy of classes: the model is analogous to inheritance hierarchies in object-oriented modeling. This hierarchy is displayed as a collapsible tree in the "Classes" tab in Protégé-2000, as shown in Figure 1. It is easy to add and delete concepts from this hierarchy using the "C" and "X" icons shown in the upper right corner.

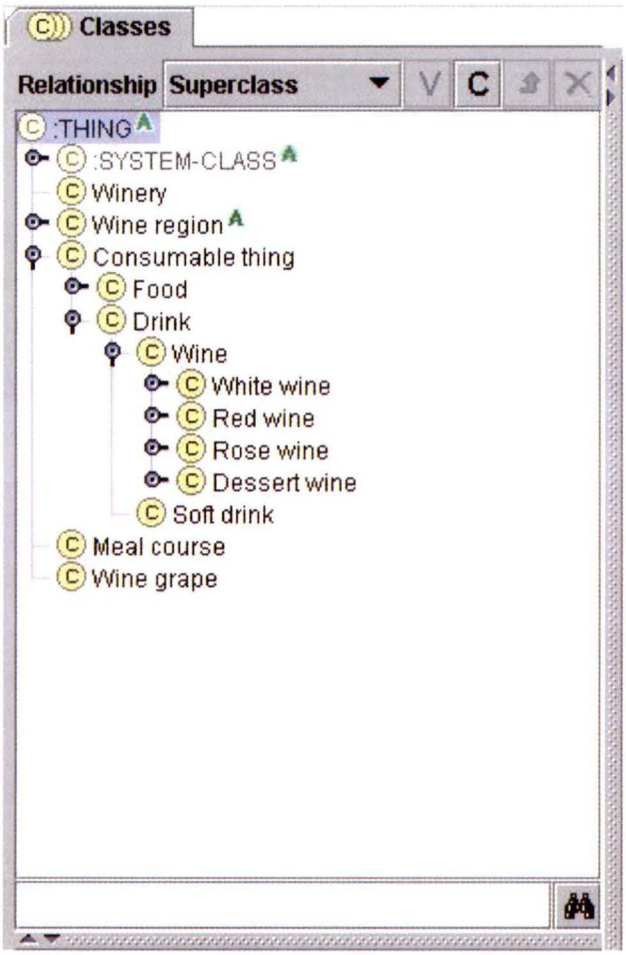

Figure 1 - The Protege-2000 Classes Tab, displaying the class hierarchy for the "wines" knowledge base.

The properties of these concepts are modeled as *slots*: in the wines example, the "wine" class will have slots for its colour, body and flavour properties, and all three types of wine will inherit these slots since they also have these properties. We can add slots to concepts at lower levels of the hierarchy as they are distinguished from other concepts: for example, the "red wines" class has a slot for "tannin level" that is not attached to the white or rosé wine concepts. Slots can have values that are text strings, numbers, boolean flags or enumerated types, but they can also have values that are other classes or instances in the ontology. For example, the "winery" class has a slot called "produces" whose values are members of the "wines" class.

Once the concepts of interest and their properties have been described in an ontology, the final step is to instantiate the concepts or to create instances. A specific wine, such as "Château Morgon Beaujolais", is an instance of a type of wine, the class "Beaujolais" (which is a subclass of "red wines"). The values for each of that class' slots must be filled out. Protégé-2000 facilitates this knowledge acquisition step by automatically generating forms based on a class' slots and their allowed values. Figure 2 shows the completed Protégé form for the "Château Morgon Beaujolais" instance.

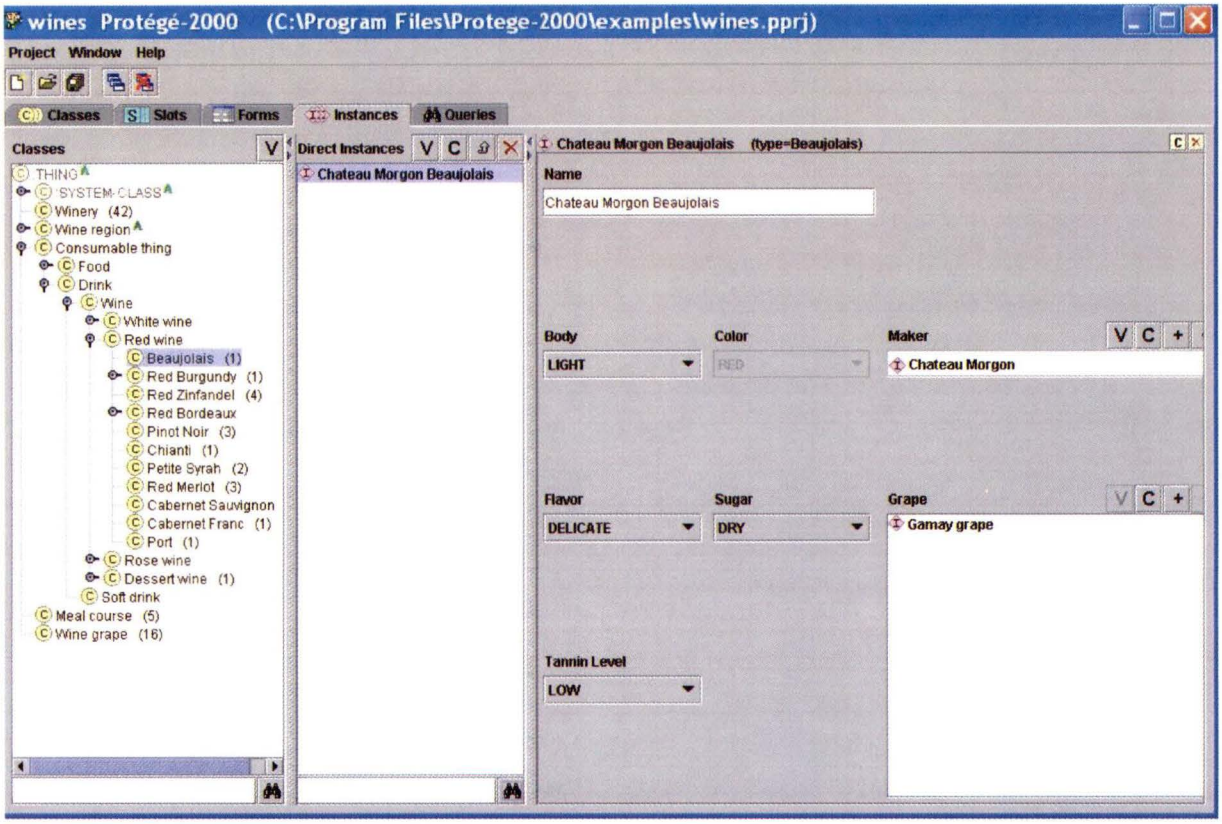

Figure 2 - A screenshot of the Protege-2000 interface, showing an automatically-generated form for a particular wine in the Instances tab. The slot values have been filled out by a human expert.

2.2 History of Knowledge-Based Systems

The first computer systems built to support decision-making were developed as early as the 1960s, using decision theory and probability theories to assist users. The first systems known as expert or knowledge-based systems (KBS), however, were not developed until the late 1970s. The distinguishing features of these systems were that they encoded expert knowledge about a problem domain, and employed some heuristic reasoning algorithm to apply this knowledge [6].

These first expert systems were usually only applied to 'toy' domains, and failed in many cases to scale to industrial applications. The need for the discipline of knowledge engineering arose out of the efforts of the Artificial Intelligence community to develop knowledge-based systems on a scale that would be useful in industry. Studer *et al.*[6] characterize the goals of this new field as "changing the construction of knowledge-based systems from an art to an engineering discipline". They draw the analogy between the

knowledge engineering community's goals to develop "appropriate methods, languages and tools specialized for developing KBSs" to the software engineering community's goals of formalizing the software development process.

The traditional approach towards knowledge engineering (KE), seen in the early 1980s, was that of a "mining" or "transfer" activity: knowledge was "extracted" from available sources such as textbooks and experts, and subsequently "transferred" to the target system [6,10]. This approach had limitations: it was difficult to maintain the resulting knowledge base, and resulting systems were unable to explain their behaviour in a way that was understandable for humans. Furthermore, it was unclear how one could re-use parts of a system for a new application. According to Schreiber [10]:

> "...this transfer approach was only feasible for the development of small prototypical systems, but failed to produce, large, reliable and maintainable knowledge bases."

As a result, the second half of the 1980s saw a shift from the "mining" paradigm to "model-based" knowledge engineering. Schreiber [10] describes the two distinguishing features of this new paradigm:

- **Implementation-independent description of knowledge**: Schreiber describes an intermediate level of description between the description of data elicited and the final representations in an operational system. The KE community has adopted the convention of calling this the "knowledge-level model".
- **Knowledge engineering as a constructive activity**: Instead of trying to replicate cognitive processes used by experts in a domain, knowledge engineers started to concentrate on constructing models that could achieve comparable results to experts on problem-solving tasks. Knowledge acquisition was no longer seen as a knowledge transfer process but as a model building process.

This modeling paradigm has allowed the latest directions in knowledge engineering, including the Semantic Web project led by the World Wide Web Consortium [11]. The goal of this project is to make information available on the Internet understandable to computers as well as to humans, to allow intelligent searching and indexing for easier navigation. As knowledge bases become larger and less navigable, new automated and semi-automated ways of intelligently searching through these spaces become more crucial.

2.3 Empirical Evaluation in Knowledge Engineering

By and large, consensus continues within the knowledge engineering community that the process of building knowledge-based systems is a modeling activity, though Shadbolt [12] admits there are dissenters, even on this point. Greater controversy exists over the best methodology or framework for performing this modeling, exactly what information should be modeled and how, and the best tools for the job. Hence the field of knowledge engineering has seen the need for evaluation methods that reflect real practice. Cohen's "Empirical Methods for Artificial Intelligence" [13] is widely cited as a seminal work in this area. However, the field has yet to reach a stage of maturity at which standard, accepted methods and frameworks for evaluation exist [14].

2.3.1 Challenges in Empirical Evaluation in Knowledge Engineering

There are many challenges that prevent researchers in this field from conducting evaluations and establishing standard methods. Shadbolt [12] notes the following challenges to empirical studies in knowledge acquisition:

1) *Amassing enough experts to provide statistical significance*
   Knowledge engineers and domain experts are usually busy, highly trained professionals. In the rare case that enough possible study participants can be found in the same context, they are frequently too busy to participate in user studies or experiments. Approaches to empirical evaluation have included using smaller numbers of experts, or perhaps even a single genuine expert. Another approach is to use near-experts or students, though whether students can be considered representative users is still a matter under investigation. Finally, some experimenters have attempted to use trivial but rich domains in which everyone is an expert, such as fruit classification.
2) *The lack of a 'gold standard' of knowledge*
   There is often no standard against which elicited knowledge can be evaluated for completeness or correctness. While it is possible to model knowledge incorrectly, there is not necessarily only one correct way to model knowledge in a particular domain; and the expertise required to judge a knowledge base describing a field is often scarce.

3) *The lack of generalizability of results*

   Studies in a particular domain are hard to generalize to other areas without enough evaluations to perform a meta-analysis. Experiments should therefore be designed not only within a domain but across several, to assess a tool's value across domains.

4) *Difficulties in separating the effects of tools, models and frameworks*

   Many areas of software evaluation must deal with the challenge of determining if the poor performance of a tool is the result of a weak tool or of a weak user interface. These problems are compounded in this domain, however, as many knowledge engineering tools are part of a suite and possible "interference" from other tools may affect results.

5) *Difficulties in quantification of knowledge*

   In many cases, it has proven difficult to know what one should measure in an evaluation, and how to measure it. While the quantity of knowledge acquired from an expert can seemingly be compared, variations in the quality of the knowledge acquired, or the value of that knowledge, mean this task is harder than it seems. A small amount of knowledge may be very important in terms if its inferential power, and an approach's ability to gain a small piece of knowledge not available by other means may be very valuable. Other possible measures include gain-for-effort, the level of human understanding achieved, the time required to accomplish certain tasks, or users' rates of error-detection.

The field is also rife with other challenges. As Tallis *et al.* note [15], the costs associated with empirical evaluations are prohibitive. Evaluations require time and effort, and comparative studies may produce unfavorable results for one's tool.

To gain insight into how these challenges might be overcome in our own evaluation, the next sections provide a review of the techniques used in previous evaluations in this field. This literature review will provide the following categorization:

1. Evaluations of frameworks or methodologies
2. Evaluations of models
3. Evaluations of tools

2.3.2 Evaluation of frameworks and methodologies

As previously mentioned, there are many different approaches to building knowledge-based systems. This is an unsurprising fact considering the large range of application domains. According to the web site for Knowledge Acquisition Initiative of the Knowledge Acquisition Community[1] [16], methodologies

"provide methods and tools that systematically extract and model knowledge used for building knowledge-based systems or solving other knowledge management problems."

Some of the most established methodologies currently in use include CommonKADS [17], MIKE [18], VITAL [19], EXPECT [20], and Protégé-2000[8]. A discussion of the details of these methodologies is beyond the scope of this thesis; however, a discussion of the techniques used to evaluate them is relevant.

Early evaluations of methodologies include the Oak Ridge Study conducted in 1983, experiments conducted in the late 1980s and early 1990s by Nottingham University and its partners called the Alvey and ESPRIT ACKnowledge projects, and the first two "Sisyphus studies" carried out by the European knowledge acquisition community in the early 1990s [12]. Each of these studies took a similar form: a group of researchers agreed to develop systems for some common problem. The Sisyphus experiments tried to evaluate these frameworks across several domains by picking different problem domains for each experiment; they did, however, all involve the same kinds of analytic tasks. These efforts created benchmarks that teams strived to meet, but no actual empirical evaluation took place. Evaluation was subjective, done by creators of the tools, and no formal metrics or measures of amount of effort required by an approach were put in place. With few exceptions, participants reported a single run of the system in their results. Furthermore, the experiments involved an unrealistic KA process since the knowledge was provided, not elicited from experts. Shadbolt [12] notes:

"Sisyphus I and II have been important prerequisites for experimental evaluations [...] but did not in themselves constitute evaluations."

---

[1] This is an interesting project, aiming to create an ontology that models the Knowledge Acquisition Community's knowledge of the domain. They wanted to test creating ontologies collaboratively, in a distributed setting.

Sisyphus III was more experimental in design than its predecessors. First, the community identified criteria that would be used to objectively evaluate and compare the approaches. Second, a wider variety of knowledge acquisition techniques were applied to a panel of experts in the chosen domain. The fourth Sisyphus experiment, following an obviously different tack from the previous experiments, intended to

> "encourage projects' collaborative use of tools through the net and web and by the integration of web tools at different sites throughout the net" [21].

The latest Sisyphus effort, called the "High Quality Knowledge Bases Initiative", focuses on the reliability and quality of knowledge-based systems, and asks participants to follow the same software process and submit their resulting systems to independent assessment. In the latest call for proposals for this latest Sisyphus effort, Menzies [21] writes that, despite pessimism that none of the earlier Sisyphus experiments have yielded much evaluation information, the researchers involved remain optimistic and the project continues.

While the European Knowledge Acquisition community was focusing on Sisyphus, another large-scale effort involving the evaluation of knowledge engineering methodologies was begun in the United States. The High-Performance Knowledge Bases (HPKB) Project aimed to develop new technologies for developing knowledge-based systems more rapidly [22]. "Challenge problems" were developed to test the tools and techniques that groups were developing. Three criteria of these "challenge problems" were that they must be challenging, they must require little or no access to subject-matter experts, and they must have unambiguous criteria for evaluating solutions. Two integration teams each solved three of these problems, and an experimental comparison of the solutions was conducted in two phases. In the first phase, similar problems to those used during system development were used and performance data collected. In the second phase, the problems required significant modifications to the systems. Both systems were tested at the beginning and the end of each phase, with the aim of measuring the ability to change the systems.

The "challenge problems" created for HPKB continue to be used, and more are being created that are increasingly challenging as a part of the ongoing DAML (DARPA Agent Markup Language) Experiment 2002.

Menzies writes that

"Despite several major international collaborative projects...little has been concluded about the relative merits of different [knowledge acquisition] techniques."[23]

In [23], he reviews data points from the evaluations mentioned here to discuss open questions and issues in the field.

2.3.3 Evaluation of models

Menzies [21] claims that the underlying models used by knowledge engineering methodologies are made of "primary modeling constructs": *General Inference Engines, Axioms, Ontologies, and Libraries of Problem Solving Methods*. Assessing and comparing the use of these constructs is different from evaluating the knowledge bases or knowledge-based systems that use them. Gomez-Pérez [24] provides a review of methodologies and techniques used for assessing knowledge-based systems themselves; however, there is very little work assessing the models involved in the knowledge engineering process. Vega *et al.* note that:

"Although Software Engineering and Knowledge Engineering provide detailed features for evaluating and assessing Software Engineering and Knowledge Engineering products...the literature reviewed in the field of ontologies shows that there are few papers about identifying features for describing, comparing and assessing ontologies" [25].

Similar claims can be made about the lack of evaluation for other modeling constructs.

Vega *et al.* further put forward a taxonomy of over 70 features of ontologies for the purposes of categorization and comparison, while Jasper and Uschold [26] provide a framework for the classification of ontology applications to describe the ways in which ontologies are used. While these kinds of works mark the beginning of investigations into assessing the quality of the modeling constructs used and provide principles for guidance in this area, there are no real empirical results involved.

2.3.4 Evaluation of tools

As previously mentioned, it is difficult to separate out the effects of a methodology, the underlying model, and the tools used in the above-mentioned experiments. While the previous studies focus on comparing the underlying processes and models, not the tools themselves, evaluators in the Nottingham and Sisyphus experiments admitted that they knew the tools and their usability were affecting the results [12].

The explosion of tools for knowledge base related tasks in recent years points to a need for standard methods to evaluate these tools, independently of the underlying methodology they are built to support. Recently there have been more and more tools built for each methodology, each with a specific and narrow purpose. The Protégé system, for example, is a tool for authoring ontologies and guiding knowledge acquisition from domain experts. Through its plug-in facility [27] it has allowed the development of many complementary tools that can operate on knowledge bases created by the system: plug-ins exist for visualizing, understanding and maintaining knowledge bases, merging or dividing knowledge bases, converting them from other formats, or working on them collaboratively.

Despite this explosion of tools, little attention has been paid to their independent evaluation. Exceptions include Duineveld *et al.*'s "Wondertools: A comparative study of onotological engineering tools" [28]which evaluates and compares several tools for building ontologies including Protégé. The authors found that the usefulness of the tools involved depends on the users' levels of expertise and the stage of ontology development. Other exceptions include empirical evaluations carried out of Protégé-related tools by the Protégé team at Stanford. These include user studies evaluating the use of domain-specific tab plug-ins for Protégé [14], and of a tool for interactively merging ontologies called PROMPT [29]. In their evaluation of PROMPT, four users were asked to merge two ontologies using the tool, and the resulting ontologies were compared to evaluate the quality of the suggestoins made by the PROMPT tool. The study of the domain-specific tab was performed as an ablation experiment, in which users are asked to perform knowledge-engineering tasks both with and without the tool under investigation. The authors found that subject matter experts were able to use Protégé with only 1-2 hours of training, and entered knowledge correctly overall 94% of the time. They also concluded that the domain-specific tab greatly enhanced experts' ability to identify known errors in a knowledge base. Protégé-2000's plug-in architecture facilitates this 'ablation experiment' approach, since through it experimenters are able to easily add or remove the tool under investigation from the knowledge engineering environment.

Chapter 3: Information Visualization

In this chapter we examine the field of information visualization to understand its potential contributions, and the evaluation techniques practitioners in this field have used.

3.1 What is Information Visualization?

Humans have inherent limits to their information processing abilities. According to Norman [30], "The power of the unaided mind is highly overrated. Without external aids, memory, thought and reasoning are all constrained." Consider our need for diagrams to evaluate design decisions or for at least a paper and pencil for all but the simplest mathematical calculations. Such cognitive aids that exploit our strengths and compensate for our weaknesses have been used for centuries. However, advances in computer technology have allowed for powerful new cognitive aids to be created. The field of computer-based information visualization aims to create software tools that exploit the human visual system to help people explore or explain data [31].

Card, Mackinlay and Schneiderman define information visualization as "the use of computer-supported, interactive, visual representations of abstract data to amplify cognition" [32]. The field of information visualization explores how these visual representations can best take advantage of humans' inherent cognitive and spatial abilities, to help them better understand abstract information. The field itself, like that of knowledge engineering, is also young. Advances in this field have only been possible since the absorption of advanced interactive graphics capabilities, such as geometric transformations in 2D and 3D, into mainstream personal computer platforms in the early 1990s [32].

Card *et al.* also propose that there are six major ways in which visualizations can amplify cognition:

1. by increasing the memory and processes available to users,
2. by reducing the search for information,
3. by using visual representations to enhance the detection of patterns,
4. by enabling perceptual inference operations,
5. by using perceptual attention mechanisms for monitoring, and

6. by encoding information in a manipulable medium [32].

3.2 Evaluation in Information Visualization

While empirical evaluation techniques examine the use of a tool by its intended users, the creators of a system can apply analytic evaluation methods by careful reasoning and analysis. In this section we examine analytic and empirical evaluations used to date in this field.

3.2.1 Analytic Evaluation

According to Card *et al.* [32],

"Initially, the exploration of effective visualizations was primarily intuitive, informed by the principles developed for static presentations and resulting in a wide range of promising designs..."

The authors point out that experience so far in this field has led to the beginning of identification of principles for the design of effective visualization that can be used for analytical evaluation.

In visualization, underlying data can be mapped to visual representations in many possible ways. A possible mapping can be evaluated in terms of its ***expressiveness*** and its ***effectiveness***. A mapping is said to be expressive if all the data, and only the data, are also represented in the visual representation: that is, that information is neither lost, nor incorrectly implied, in the mapping. A mapping's effectiveness, on the other hand, is the degree to which it is "faster to interpret, can convey more distinctions, or leads to fewer errors" [32] than another mapping.

In [33], Card and Pirolli propose a "Cost-of-Knowledge Characteristic Function" in which the amount of information that can be reached is compared to the cost of reaching that knowledge – a cost usually measured in time. Furnas also explored this idea of effectiveness [34], deriving the heuristics that "despite the vastness of an information structure, the views must be small [presenting only a small subset of the structure], moving around must not take too many steps and the route to any target must be discoverable" for a visualization to be effective. In [35], Tweedie sought to characterize the space of interactive techniques for visualizations with respect to the types of data represented, the nature of the visible feedback displayed, and the forms of interactivity

used. These types of characterizations can also serve to compare different visualizations and evaluate potential designs.

3.2.2 Empirical Evaluation Techniques

Much of the empirical evaluation in this field has taken the form of comparative studies that borrow techniques from the field of human-computer interaction such as Goals Objectives Models Solutions (GOMS), as in "A Table Lens Tool for Making Sense of Data" by Pirolli and Rao [36], and user studies or experiments. Several examples of comparative user experiments of well-known techniques can be found in a special issue of the International Journal of Human-Computer Studies devoted to the issue of empirical evaluation in information visualization [37]. However, in the guest editorial for this issue, Chen and Czerwinski report,

> "We are finally beginning to see evaluations of advanced visualization techniques, but progress has been slow and isolated" [38]

In Chen and Yu's meta-analysis of empirical evaluations in this field [39], the authors conducted an extensive search of on-line resources, such as the ACM Digital Library and online IEEE conference proceedings. Their search located only 35 experimental studies published between 1991 and 2000, 32 of which were published between 1995 and 2000. They conclude that there has not yet been enough empirical work in this field for a statistically significant meta-analysis.

The existing empirical studies usually try to demonstrate whether an innovative interface supports the intended users better than another interface in a given task. Therefore, a key prerequisite for useful evaluation is an understanding of the tasks under consideration. As mentioned by Card *et al.* [32],

> "The key to understanding the effectiveness of information visualization is understanding what it does to the cost structure of a task. Depending on the task, visualization could make a task better – or it could make the task worse."

3.3 Information Visualization in Knowledge Engineering

3.3.1 Visualization Techniques and Tools for KE

The use of information visualization techniques remains quite limited in knowledge and content management systems, but exceptions include SemNet [40] and more recent

techniques based on the AT&T Graphviz tool [41] or on the Touchgraph graph layout algorithm [42]. GraphViz is a somewhat limited tree-viewer, in that it provides no interaction mechanisms and provides poor scalability to large knowledge bases. Examples of GraphViz-based tools include Ontoviz[43] and IsaViz [44]. While these visualizations provide poor interaction facilities they are excellent tools for creating static diagrams of ontologies, such as the diagram of a part of the wines ontology in Figure 3.

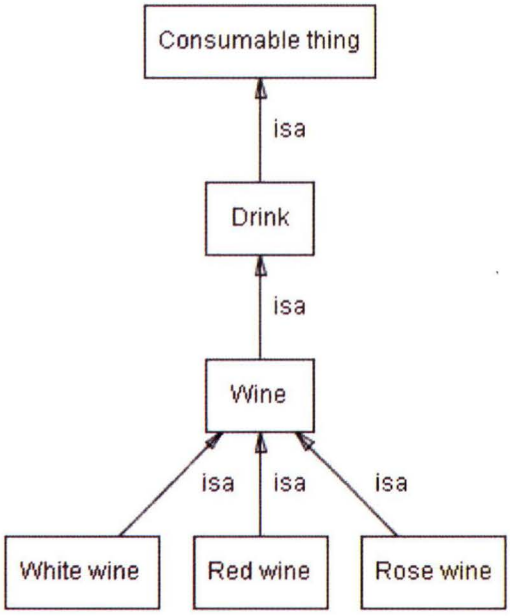

Figure 3 - A part of the "wines" example knowledge base represented in the Protege-2000 Ontoviz tab

The TGVizTab for Protégé [45] and the native visualization in the Kaon tool [46] are based on the Touchgraph algorithm. The TGVizTab allows the user to specify a concept of interest and incrementally explore related concepts through relationships. Figure 4 shows a screenshot of the same part of the wines ontology represented in the TGVizTab.

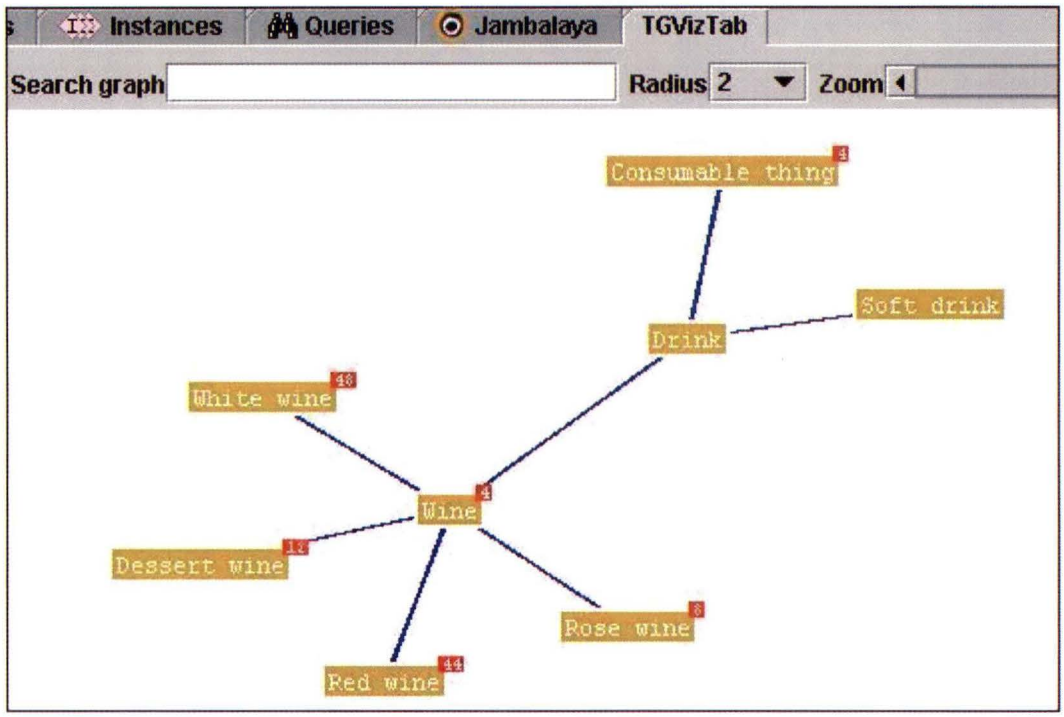

Figure 4 - A part of the "wines" knowledge base represented in the TGVizTab

Chaomei Chen's work in the visualization of large knowledge bases in the academic publication domain [47,48] is relevant to the introspective case study in Chapter 6, which also explores managing knowledge concerning academic documents. He makes use of visualization to explore large document spaces and to help users find relevant academic literature. His work centers on the visualization of co-citation indexing. Using citation as a measure of 'expert' opinion of a document, co-citation looks at the number of times a document is cited with other important works. Applying these visualizations to large document repositories such as Citeseer can reveal the more important authors or papers in an area.

Related work also includes "Interactive Visualization Techniques for Ontology Development" by Ng [49] in which the author discusses the tasks performed in authoring, verifying and exploring knowledge bases in a description logic system. While the reasoning tasks involved in a description logic-based system are somewhat different from those in frame-based systems such as Protégé, many of his findings are very relevant to this work. He identifies the key role of browsing a knowledge base in verifying two kinds of consistency: domain and model consistency. He defines domain consistency as the agreement between the domain and how it is modeled in the knowledge base, and

model consistency as "consistent understanding and usage of terms among modellers to avoid redundancy and conflicting definitions." He also finds, through user interviews and work observations, that the textual knowledge base browser he examined failed to support browsing tasks that involve the examination of multiple concepts. He also proposes visualization techniques to support these tasks in a tool called InfoLens, based on the idea of Xerox's Lens filters [50]. It uses a system of overlapping lenses in which each lens acts as a data-filtering component. A stack of lenses allows an arbitrary subset of data to be gathered and formed, and the resulting information is visualised as a data table.

3.3.2 Jambalaya

Through Protégé's plug-in facility, developers can easily add new component functionalities to the Protégé tool [27]. Over the past three years, members of the CHISEL group have been developing and integrating a sophisticated interactive visualization tool with Protégé called Jambalaya. We created the Jambalaya plug-in by customizing and extending the SHriMP Views general-purpose visualization framework developed by our group at the University of Victoria [4]. SHriMP includes generic visualization techniques and navigation facilities to make better use of people's cognitive abilities when using a screen of limited size. SHriMP uses a nested graph view to present information that is hierarchically structured. It combines a hypertext following metaphor with animated panning and zooming motions over the nested graph to provide continuous orientation and contextual cues for the user.

In Jambalaya, concepts and relationships are presented using the graph metaphor. Classes and instances are represented as nodes in a graph; different types are distinguished using color. Directed edges (arcs) are used to show relationships between concepts and instances, such as *is-a* relationships between classes in the concept hierarchy, *instance-of* relationships relating instances to classes, and s*lot relationships* between classes and instances. Figure 5 depicts how a part of the wines ontology might look in Jambalaya; in this figure, the nodes represent concepts and the arcs represent the 'is-a' relationship.

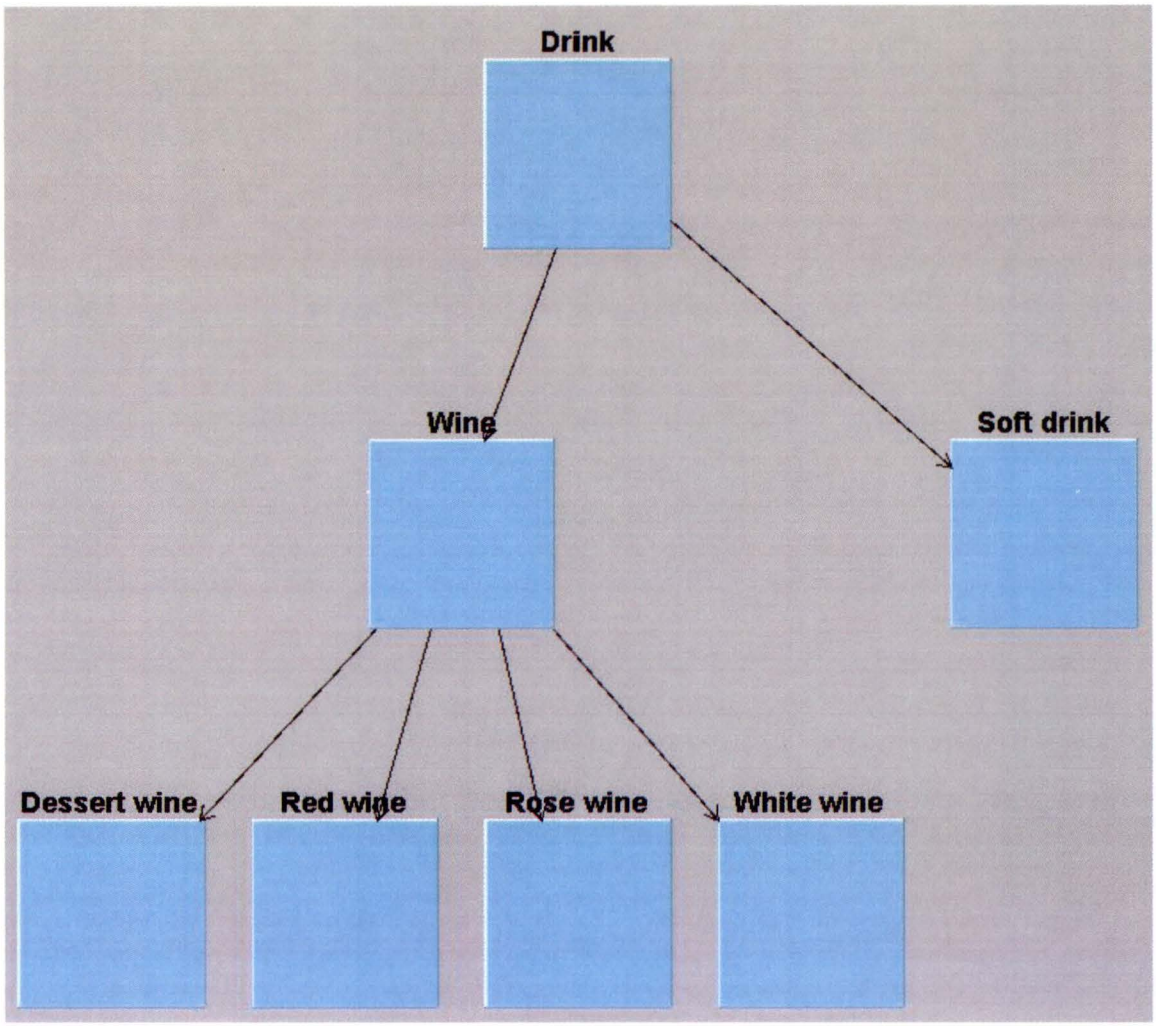

Figure 5 - The "wines" ontology represented in Jambalaya. In this case, each node represents a Protégé class, and the arcs represent the "is-a" relationships between them.

In addition to using arcs to show relationships as connections between classes and instances, the user may show a relationship type using containment. For example, rather than drawing arcs between nodes to show the *is-a* relationship, we can instead nest subclasses within their superclasses. Figure 6 again depicts a part of the wines knowledge base in Jambalaya; however, in this figure the nodes are nested on the 'is-a' relationship. Furthermore, it may also be advantageous to use other user-defined slot types for nesting nodes. For instance, in an anatomy ontology, the user-defined *part-of* relationship may be a more appropriate relationship for nesting nodes than the *is-a* relationship.

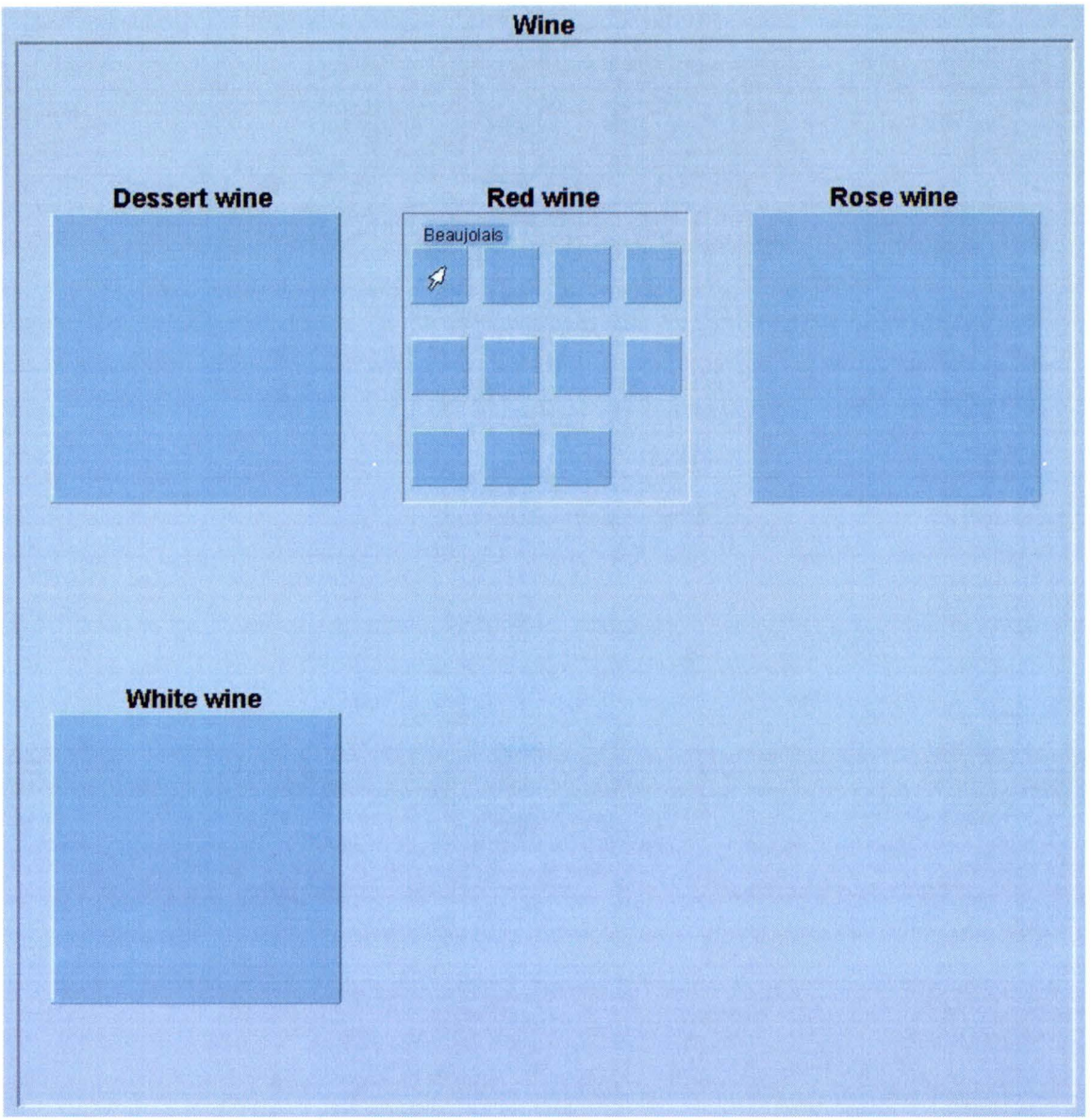

Figure 6 - Another representation of the wines ontology in Jambalaya. In this figure, the nodes are nested within their parent nodes, based on the "is-a" relationship.

The Jambalaya tool includes many features and capabilities that have evolved as the tool was developed. Many were included in the tool based only on a team member's intuition that they would be useful or a desire to examine the issues involved in implementing such a feature, with insufficient knowledge of their utility or how frequently they would be used in practice. The tool's interface has therefore become more complicated as the functionality has increased. Figure 7 shows a screenshot of the basic Jambalaya interface.

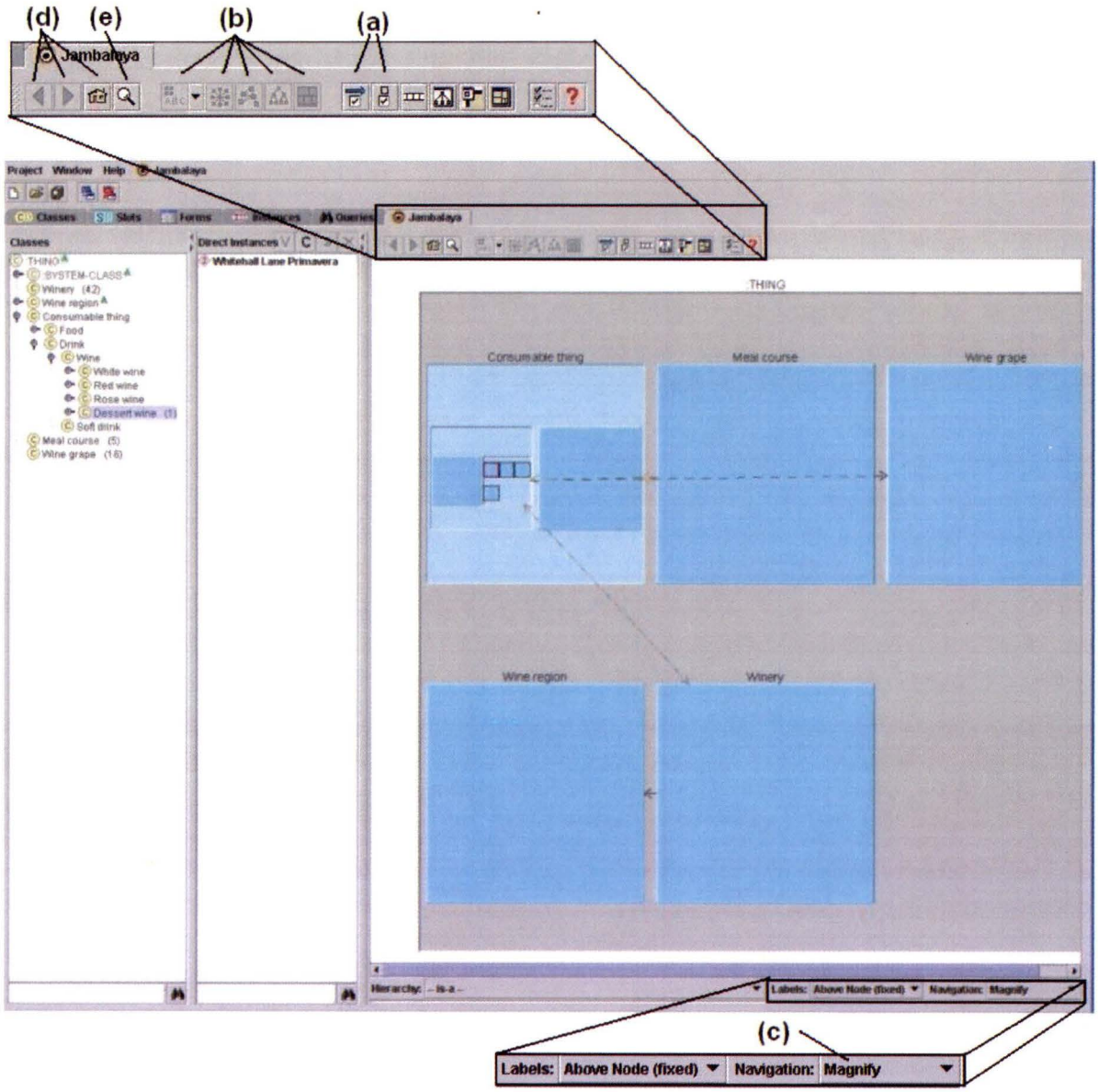

Figure 7 - A screenshot of the Jambalaya interface, with magnified views of toolbars

In the default configuration for Jambalaya, the interface is even more complex than in Figure 7, as it includes several floating panels relating to presentation, navigation and collaborations features.

Presentation features include arc and filter palettes that allow users to decide the relationship and concept types to be displayed by selecting these types using checkboxes (see Fig. 7a). They also include options to apply several automatic layout algorithms to the visible nodes and arcs, including grid, radial, hierarchical, force-directed or "spring" and treemap layouts (Figure 7b).

Users have several options for navigation (Figure 7c). They can zoom in and out from the nested diagram using one of three modes: magnify, which magnifies a selected node to fill the screen; zoom, which allows the user to zoom incrementally on any part of the canvas; or fisheye zoom, which enlarges a selected node to show more detail while shrinking the surrounding nodes to maintain context for the user. The interface also follows a hyperlink-following metaphor in that users can click on any slot values in a node's form to navigate to the associated concept in the hierarchy. It includes browser-style navigation buttons (back, forward and home) as shown in Figure 7d. Users can navigate the knowledge base using the collapsible-tree view familiar from the Protégé interface shown on the left of Figure 7; this view is coupled with the zoomable canvas so users can switch navigation styles easily. Finally, users can simply navigate using a text search option to find particular concepts (Figure 7e).

Many orientation cues are offered to the user while they navigate a knowledge base, including tool tip labels that are displayed when a user holds the mouse cursor over a node, the hierarchical view which displays the 'is-a' hierarchy in a smaller, secondary pane, and a thumbnail view which provides an overview of the whole canvas and allows users to drag or re-size the current view in a different part of the canvas. The user also has several options to change the appearance of labels on each node.

Members of the CHISEL group have recently begun examining support for collaborative activities in knowledge engineering. To support sharing a person's understanding of a knowledge base, the filmstrip feature allows the user to save and annotate a particular 'view' of the knowledge base that can later be re-loaded by other

users. Another feature that supports this sharing is the "export to SVG" function that creates an SVG graph of the current view that can be published to the World Wide Web.

Finally, to support the varied user base and individual differences, many options in Jambalaya are customizable through the "Options Panel", including user controls, the zoom speed, and the appearance of arcs and labels.

3.3.2 Challenges to Empirical Evaluations in this Field

Empirical evaluations of visualizations that support knowledge acquisition or knowledge engineering are difficult to find in academic literature. This lack is not unusual for a young sub-field of software engineering; nor does it indicate a failure or laziness on the part of researchers in this area. As we saw in the field of knowledge engineering, formidable challenges keep researchers from performing these evaluations; indeed, information visualization tools for knowledge engineering face the same barriers to evaluation as all knowledge engineering tools, as well as some additional challenges. Expert users are already hard to assemble in a knowledge engineering context; users in evaluations of these tools must furthermore be experts in visualization techniques, or be given some incentive to learn to interact with them. While interaction techniques for visualizations can be simplified to reduce training time, there are often trade-offs between the complexity of using a tool and the power of the visualizations it provides. For sophisticated visualizations, experimental results may be affected if users are not accustomed to advanced interaction techniques, or are not given enough training and practice time with the tool. This can be difficult to accommodate given the limited time such experts can afford for user experiments.

3.3.3 Empirical Evaluations

Searches for literature in this area indicate that empirical evaluations of information visualization tools for knowledge engineering are rare, or at least seldom published in academic literature. One exception is [49] in which the author evaluates proposed visualization techniques for creating ontologies, with respect to their ability to aid pattern detection. His evaluation uses Collaborative Evaluation, a think-aloud technique that uses a small number of participants to capture general and major usability problems.

EARLY EVALUATIONS OF JAMBALAYA

Until recently, no formal evaluations of Jambalaya's usability had been performed. The development team, being familiar with HCI principles, was aware of certain glaring usability problems, but usual outside pressures kept fixes for these 'minor' problems from being performed. A heuristic evaluation [51] of the tool conducted in November 2002 raised issues that were discussed with the CHISEL group to prioritize action items and decide on future directions. As a result of this work, the interfaces for both Jambalaya and the SHriMP tool it was based on were redesigned to include a toolbar, with icons for performing most actions. Previously, many of the tools' features had only been accessible through drop-down and contextual menus.

Another evaluation of interest is a series of user studies performed using a tool based on SHriMP in a knowledge management domain. For these studies, we compared two tools for exploring and understanding business flows. The Business Flow Editor interface in IBM Corporation's WebSphere Application Developer Integration Edition tool (a sophisticated tool for web service development) uses a tabbed interface to present different levels of process nesting. The second interface was a prototype for viewing business diagrams based on SHriMP that employed a zoomable user interface, called Ebi (described by Rayside *et al.* in [52].) The primary purpose of the study was to see if there were any differences between zoomable nested interfaces and tabbed pane interfaces in browsing and understanding hierarchical business flow diagrams. We expected that zoomable interfaces would better support users trying to understanding a business flow because they provide continual reinforcement of any sub-process's context and place in the hierarchy.

The study was conducted in two phases: a pilot study involving 5 student users was conducted at the University of Victoria, and the final study involving 10 more experienced participants was conducted at the User-Centered Design Laboratory (UCD) at IBM in Markham, Ontario one week later. The evaluation procedure involved asking users to perform certain flow exploration tasks, speaking their thoughts aloud as they proceeded. Users in both phases of the study were also asked to complete several questionnaires during the experiment.

While four out of five users in the pilot study preferred the zoomable interface, seven out of ten users in the final study preferred the tabbed interface. We attribute these differences to several potential factors:

1. Larger, more complicated examples were used for final study than in the pilot study. This may indicate that the size of the business flow being examined might affect the preference of one interface over the other.
2. Subjects received more help with tools during pilot studies. This may have reduced the effects of usability problems subjects experienced with the Ebi tool during the main study. This possibility is supported by the discrepancy in usability ratings for Ebi prototype between the two studies: users experienced many usability problems in the pilot study, but these difficulties did not cause the same dissatisfaction seen in the final study.
3. Pilot study users were not compensated for their time, while final study participants were. Final study subjects may therefore have had a higher degree of investment in the study process and results, and may have taken the study more seriously.
4. During the pilot studies, tasks were more focused on asking subjects to use the features of a tool than on asking them to complete business-flow related tasks. This may have shifted the subjects' focus away from the utility of the tool to looking at the attractiveness of its features during the pilot. The task list was changed before the final study to be more focused on asking users to complete tasks as opposed to asking them to use certain features.
5. The demographics of the users changed between the pilot and final studies. Participants in the pilot were senior undergraduate or graduate students, most of whom had little domain experience, while several participants in the final study were business flow editing professionals.

This last possibility has implications for user studies that attempt to replicate 'expert' behaviour using undergraduate students or 'novice' users. Further exploration is needed to determine if this practice has an effect on the results of these studies. Factors 3 and 4 highlight the importance of understanding representative tasks in a domain when conducting user studies: potentially, the more experienced users in the final study had a clearer idea of the tasks involved in editing flow diagrams than the pilot study users.

Expert users said that they would want to switch easily and quickly between several parts of the diagram without being forced to watch the "tedious" zooming animation. While the zoomable interface seemed to provide better context and support for browsing and understanding business flow diagrams, it is possible that the tabbed interface better supports editing tasks.

An early attempt at a user study of Jambalaya, previously undertaken by the author, revealed that more exploration of the tasks involved in knowledge engineering was required before user experiments could be beneficial. In order to choose the tasks experimental users would be required to complete, the features of Jambalaya were examined for potential tasks they *could* support. This user test, performed with a novice user, helped to assess the usability of the tool, account for known usability problems in future evaluations, and guide the development of experimental and training materials for future evaluations. However, it did not succeed in providing an accurate depiction of the knowledge engineering process or how knowledge engineering or domain experts might use these tools. The user was unable to perform tasks that she had not been instructed how to do. The evaluators were simply showing her what the tool is capable of doing and assessing if the user could remember which steps to follow. While this provided good feedback as to whether appropriate visual cues had been provided to support the user in remembering these steps, it did not assess whether the processes shown to the user are in fact performed in the same manner by real users, or even used at all. A multi-user experiment at this stage would have provided an assessment of the usability, but not of the utility, of the tool.

Instead of using the features in Jambalaya as a starting point and working backwards to see what tasks they might support, further investigation into the tasks in need of visualization support was required. The next chapter details how, by reflecting on the previous evaluation approaches reviewed here and the challenges they pose, we can select appropriate approaches and techniques to develop theories and hypotheses for later verification.

Chapter 4: Empirical Evaluation and Technology Maturation

As investigations in a field mature, the questions being asked become more precise [3]. This chapter investigates the process by which this maturation happens, by looking at general approaches and techniques in empirical evaluation. It then steps back to analyze the evaluations reviewed in the previous two chapters with respect to these approaches. Taking stock of the maturity of the evaluation process in these fields is a critical step: looking at the questions that have been asked is crucial before developing questions of our own.

This examination shows that evaluation in the field of information visualization for knowledge engineering is still at very early exploratory stages; theory building in this area is still at a stage of gathering user requirements for more understanding of the problem domain. We therefore also examine the field of requirements engineering and the techniques used in this area. This motivates the approaches used in this thesis: a case study observing expert knowledge modelers in their working environment, and an introspective case study in which a knowledge management tool was designed, created, and used by our research group.

4.1 Evaluation Techniques and Approaches

According to Basili [1], the scientific method involves the following steps:

- Observe the world
- Propose a model or a theory of behavior
- Analyze
- Validate hypotheses of the model or theory
- Repeat the procedure if possible

This fits with Cohen's characterization of empirical method, in which preliminary exploratory studies yield causal hypotheses that are tested in observation or manipulation experiments [13]. In Almstrum *et al.* [3], this process is described as moving from data-driven analysis, in which techniques like interviews and questionnaires, observations, case studies and introspection play a great part, to a more theory-driven approach, which

involves testing hypotheses using techniques such as quasi-experimental studies or laboratory experiments. Other distinctions between approaches can be made as well, and are summarized from [3] for this discussion.

- Formative and summative evaluation: Formative evaluation guides the formation of a project; it is conducted as the project proceeds. Summative evaluation, on the other hand, is done at the end of a project.
- Qualitative and quantitative evaluation: These are distinguished in the type of data produced, and whether it is numerical data, which often undergoes statistical analysis, or descriptive data. Distinctions between these two types can blur, such as when descriptive data is submitted to analysis that produces numerical output.
- Laboratory and *in situ* studies: Studies are sometimes conducted in a laboratory setting to control at least some factors in the experimental environment. Field studies are conducted *in situ* to observe behavior in its natural context.
- Incremental and longitudinal studies: While incremental studies look at a phenomenon over a short period of time, they can be conducted in a series over time to look at results over a longer period, as in a longitudinal study.

4.2 Analysis of Previous Evaluations

4.2.1 Knowledge Engineering

METHODOLOGIES

According to Menzies, early efforts to evaluate methodologies like the Oak Ridge Study and Sisyphus I and II can be more aptly classified as "program watching" than "evaluation"[23]. Indeed, the lack of metrics or criteria for the success or failure of an approach indicates that these early studies were very exploratory and data-driven; however, they were beneficial in that they brought attention to the strengths and weaknesses of various approaches, and in that they unified vocabularies that might have created divisions between research groups. Additionally, they encouraged participants to improve the descriptions of their methods, and indeed the methods themselves [12].

The Sisyphus experiments intended to be conducted as a longitudinal study, moving from qualitative evaluation of methods as performed by their creators to more experimental, quantitative evaluation of the resulting systems by impartial observers.

Unfortunately, while the Sisyphus experiments continued their trend moving from exploratory empiricism towards realism and complexity, their level of participation dropped dramatically for Sisyphus III, IV and Sisyphus HQKB. This drop highlights the difficulties involved in moving to more focused comparisons of approaches; as Shadbolt notes [12], a wide community may not necessarily agree on the evaluation methods and measures to be used as the focus narrows.

The HPKB Project, on the other hand, takes a much more experimental and quantitative approach, and has managed to determine some techniques for building re-useable and modifiable knowledge bases more quickly.

MODELS

The area of evaluating knowledge modeling constructs is still at the stage of developing theories and guidelines for the desirable properties these models should have. While these may be appropriate for initial analytic evaluations, as this area matures, these theories may be verified empirically.

TOOLS

Gomez-Perez [53] refers to the early evaluations of knowledge engineering tools as "evaluations" because they look at the capabilities and technical characteristics, such as the system resources they require. For example, in the early Sisyphus experiments the tools were simply judged in their ability to create the desired system. She distinguishes these from "assessments" which look at the tools created from the user's perspective. She also notes that in general, this assessment is generally lacking in evaluations of knowledge engineering tools.

Giboin *et al.*'s paper at EON2002 [54] calls for "systemizing the scenario approach", moving from evaluation to assessment by using more techniques from the field of Human-Computer Interaction and Computer-Supported Collaborative Work. It is worth noting that the summative evaluations of Protégé-related tools previously discussed are scenario-based and focus on the real user experience, while measuring quantitative results. However, these types of experiments can only be conducted for tools designed with very specific purposes in mind: the domain-specific tab evaluated by Noy *et al.* [14] aimed to increase the amount of knowledge acquired from a domain expert, and the

PROMPT tool was created specifically for merging knowledge bases. Without a deep understanding of a tool's purpose, scenario-based experimentation is not possible.

4.2.2 Information Visualization

As mentioned in Chapter 3, it is recognized that the field of information visualization has reached a point of maturity where theories for analytical evaluation can be provided. Furthermore, although there have not been enough empirical evaluations for a meta-analysis of the field, more and more such evaluations are being conducted. Largely, these evaluations suggest starting by stating claims and hypotheses, then deciding on the users and tasks to be used in the experiment. For example, in the comparison described by Pirolli and Rao [36], the authors use the GOMS technique to compare visualizations. They start by identifying representative tasks in the domain and use GOMS to form predictions about which tool will help users perform those tasks most effectively. This presupposes that experimenters know enough about the domain under investigation to select representative tasks. Such is not the case in the domain of knowledge engineering.

4.2.3 Information Visualization for Knowledge Engineering

With very few exceptions, detailed descriptions of the tasks performed by knowledge engineers and domain experts in creating and using knowledge bases are not reported in the literature. Exceptions include observational case studies such as Ng's [49] and Tallis *et al.*'s description of tasks they observed users performing during user studies [15]. The tasks they identify include:

- Browsing a KB for understanding
- Searching a KB to find a particular concept
- Editing to create or modify a KB element
- Checking that a modification had the expected effects on the KB
- Checking a KB for errors
- Understanding and deciding how to fix an error
- Undoing a previous step to correct or recover from an error
- "Stepping back" to reason about the KB

To start developing theories and hypotheses about which knowledge engineering tasks Jambalaya supports, a more data-driven approach is required to find out more about the users and the tasks involved. Without exploration of the tasks users want to perform, experiments may produce valid but inconsequential results. As an example, imagine the following scenario in a user experiment. The experimenter could ask the user to complete a task that had been demonstrated during training. If the user completes this task successfully, the experimenter may conclude the tool provides appropriate visual cues to help the user remember how to perform this action. What he cannot conclude is if this functionality is ever used in a real-use situation, or if a different interface would better support the operation.

Instead of starting with claims and hypotheses, then, the first step in evaluating Jambalaya is a requirements elicitation task to answer the following questions: *Who are the users? What tasks do they want to perform? Which of these tasks are cognitively challenging and can benefit from visualization support?*

4.3 Requirements Engineering

Gause defines development as "the process of transforming someone's desires into a product that satisfies those desires" and requirements engineering as "the part of development in which people attempt to discover what is desired" [55]. It is not a process specific to software development, but part of the creation of any new system; however, this thesis will focus on the requirement process in software system development. Nuseibeh and Easterbrook define software systems requirements engineering as the process of discovering the purpose for which a system was intended, by identifying stakeholders and their needs and documenting them in a form that is amenable to analysis, communication, and subsequent implementation [56].

4.3.1 The Field of Requirements Engineering

Requirements Engineering (RE) for software systems emerged as a field of study in its own right in the early 1990s, as witnessed by the emergence of two series of international meetings and the establishment of an international journal on the subject. It is an interdisciplinary field; while areas such as logic, systems engineering, and computer

science provide tools and methods for assessing whether a desired system is feasible and for building the software system in question, RE involves much more. According to Nusibeh and Easterbrook [56], one of the radical new ideas that have emerged since the early 1990s is that the organizational and social contexts in which the new system will operate must be taken into account during requirement modeling and analysis. Areas such as the cognitive and social sciences have much to contribute in helping to understand the human contexts in which these software systems are built and used.

In the waterfall model of system development, as depicted in Fig. 8, the requirements phase is positioned before design. However, the feedback loops in this model indicate that each activity may occur more than once, and indeed information that affects the requirements may be discovered at later stages of the lifecycle [57]. In practice this is often the case, and therefore RE plays an important role in the management of change in software development. Nevertheless, the bulk of the effort in RE generally occurs early in the lifetime of the project, motivated by research demonstrating that early, undetected errors are costly over the project lifecycle [56].

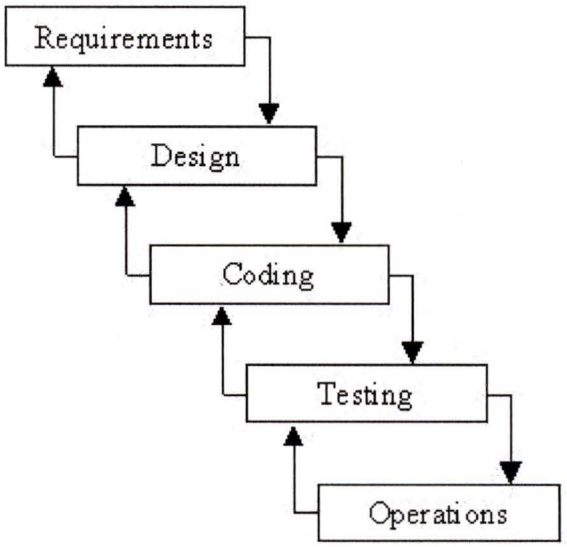

Figure 8 - A representation of the waterfall model of software development

It is worth discussing that the development of software tools in an academic context is very different from industrial development efforts. Commercial businesses generally try to create systems to fulfill a particular need; they generally do not start the

construction of a system before the need for it is assessed. Research projects, on the other hand, are more exploratory and entire systems can be built on the intuition that it will be useful – or just to see that such a system can be built at all. In academia, a tool can be built long before its specific purpose is decided; this holds the advantage that developers are able to present a working prototype to potential users for evaluation and feedback. This development context more closely resembles a software lifecycle like the spiral model [58], depicted in Figure 9.

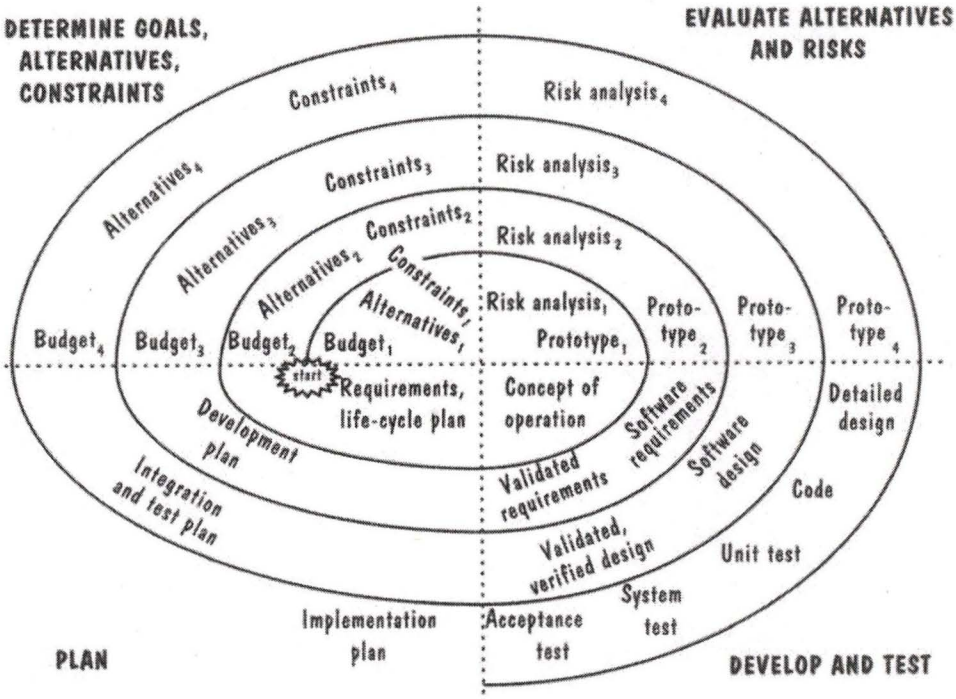

Figure 9 - The spiral model of software development[2]

While academic projects do not necessarily have the same emphasis on risk analysis, their development often follows the iterative process of determining alternatives, developing and evaluating prototypes and refining tools based on these results shown in the spiral model. To determine alternatives, however, we must first elicit the goals of the system.

---

[2] Diagram taken from public domain

4.3.2 Requirements Elicitation Techniques

There are many requirements elicitation techniques that have been found to be useful in various situations, each with their own particular strengths and weaknesses. Broadly, they include [57,59]:

- Traditional techniques such as interviews, questionnaires, and surveys. Closed or open-ended questions can be asked in interviews, and they have the benefit of allowing the analyst to probe further in non-nominal cases and gather more detailed information. Questionnaires and surveys, on the other hand, are easier to distribute and to administer, but result in less rich data.
- Cognitive techniques originally developed for knowledge acquisition such as protocol analysis, laddering, card sorting, and repertory grids. In [59], the authors warn that these techniques might not necessarily give accurate depiction of what the user is really thinking: think-aloud techniques like protocol analysis, for example, have been shown to give inaccurate models of user needs. These are generally used for eliciting such things as classification knowledge or the hierarchical structure of a domain.
- Contextual techniques, which emerged in the 1990s as an alternative to both traditional and cognitive techniques. These include ethnographic techniques such as participant observation, or ethnomethodology and conversation analysis (which apply fine-grained analysis to identify patterns in conversation and interaction). These techniques can be used at early stages of software development to understand a problem domain, or later on to answer more focused questions. They are often used to gather tacit domain knowledge (which is hard to elicit by other means) or knowledge that is difficult to describe in words, in either traditional methods or cognitive, "think-aloud" type scenarios. An example of such knowledge is the process of tying a shoelace: the process is much easier to demonstrate than it is to describe.
- Model-driven approaches, which provide a specific model of the type of information to be gathered. These approaches are usually scenario-based: for each goal, the scenario of possible use is described and modeled.
- Group elicitation techniques include group brainstorming; focus groups in which props, such as prototypes, storyboards, or scenarios are used to communicate customer needs; and Joint Application Development (JAD) or Rapid Application

Development (RAD) meetings, which focus on consensus building with an unbiased facilitator.

- Introspection: without access to communication with customers, it is possible for analysts to reason about a problem and decide what is needed. However, this approach runs the risk of being highly inaccurate in deciding actual, real-world customer needs.

4.3.3 The Elicitation Process

In [60], Kotonya and Somerville maintain that there are four dimensions to requirement elicitation: application domain understanding, problem understanding, business understanding, and the needs and constraints of the stakeholders. According to [56], some of the important items to elicit during this phase include system boundaries, involved stakeholders, system goals, and the tasks the system will support.

To satisfy each of these dimensions, the process of requirement elicitation is presented as the following steps:

1. Objective setting: High-level system goals and boundaries are decided.
2. Background knowledge acquisition: This step involves acquiring background knowledge about the application domain, the problem itself and the business needs the system will satisfy.
3. Knowledge organization: In this step, stakeholders are identified and their goals are prioritized.
4. Stakeholder requirements collection: This final step is the one most often associated with the term "elicitation" in that requirements are "elicited" from customers.

4.4 Looking Forward: Evaluating Jambalaya

To begin determining the requirements for visualization tools supporting knowledge engineering activities, we must now carry out the elicitation process, as described in the last section, in relation to the Jambalaya tool. The first three steps will help us to determine which elicitation techniques are appropriate in the fourth step.

1. Objective setting: High-level system goals and boundaries

   At a high level, the goal of the Jambalaya system is to support Protégé users in creating and maintaining knowledge bases.

2. Background knowledge acquisition: Acquiring background knowledge about the application domain, the problem itself and business needs

   While descriptions of methodologies provide us with some of the high-level tasks performed by knowledge engineers and domain experts (such as creating an ontology, verifying the ontology, adding instances to the ontology, and modifying the knowledge base structure or content) the literature describing the more detailed tasks as actually performed in practice by knowledge engineers is sparse, as previously mentioned.

3. Knowledge organization: Identifying stakeholders and prioritizing their goals

   The stakeholders in knowledge engineering are a varied group, from knowledge engineers to domain experts to end-users of the knowledge-based systems. Knowledge engineers usually have experience modeling and representing knowledge, but do not necessarily have experience in the domain being modeled. Therefore they must rely on knowledge elicitation methods to gain information from domain experts. Knowledge engineers can also have varying degrees of experience with a particular model or a domain; there is therefore a division between expert and novice knowledge engineers, each category having different needs. The Jambalaya tool is intended to support both novice and expert knowledge engineers and domain experts trying to understand or verify a model of a given domain. However, the needs of the end users, who might be completely unfamiliar with both knowledge modeling concepts and the domain of interest, are beyond the scope of the Jambalaya tool. It is likely that customized visualizations for each domain will be of more use than one multi-purpose tool.

4. Stakeholder requirements collection

   Given the high-level questions we are investigating and the difficulty gaining access to expert knowledge engineers, an observational case study of a few expert users is appropriate in this case. Contextual enquiry techniques are proven techniques for acquiring implicit domain knowledge at an early exploratory stage

> [61]. We therefore chose to conduct a contextual observation case study. Furthermore, we wanted to generalize these results without requiring access to more experts, so we decided to conduct an introspective case study within our group. Members of our group had often experienced the role of the 'novice' knowledge engineer, trying to understand a new model or knowledge base. Through this introspective study, we wanted to build a model with which we could become more familiar, so that we could experience the role of the "expert" knowledge modeller. By using this technique we can also create a group of experts within our own lab that can be used as subjects for future experiments.

Trade-offs are inherent in any evaluation design [3]. Using these approaches, the results will not necessarily be generalizable to other domains; they will sometimes reflect subjective biases; and they will not support quantitative comparison with other tools or statistical analyses. However, given the many challenges faced in assembling expert users, and as we intend to investigate our results with further experimental studies in the future, these are acceptable and appropriate trade-offs to make at this stage.

4.5 Summary

This section has motivated our use of contextual observation techniques and an introspective case study conducted within the CHISEL group to determine the tasks performed by knowledge engineers, and the potential benefits visualization might bring to the performance of these tasks. Once we have determined these tasks we can evaluate whether the Jambalaya tool provides this support, or how it can be changed to better meet the needs of its users.

Chapter 5 describes the observational case study the author conducted at the Structural Informatics Group at the University of Washington. By observing two knowledge modellers working on a digital model of the human anatomy, the author was able to glean insights into the tasks these users perform and the challenges they face. In Chapter 6, we discuss the design and implementation of the Shrimpbib system, a document recommender system for small academic groups. By building this system for the CHISEL group, we experienced many of the issues encountered by knowledge engineers first-hand. The insights and observations gleaned in both of these endeavours

are brought together in Chapter 7, where we begin to analyze these results and to draw conclusions about tool support for knowledge engineers, including visualization support, and future empirical evaluations in this area.

Chapter 5: Observing Users *in situ*

Contextual inquiry techniques, as described in section 4.3.2 on requirements elicitation, differ from prototyping and usability testing in that, instead of iterating on a pre-existing idea they can generate new ideas and directions. They do this by placing the system designers in the customer's work context, giving them the richest possible experience and knowledge to invent from [61].

The Structural Informatics Group (henceforth SIG) at the University of Washington in Seattle, WA, is currently developing a large knowledge base called the Digital Anatomist Foundational Model using Protégé-2000. On May 27th and 28th 2003, the author visited this group to understand their knowledge base, to conduct contextual observations and interviews, to demonstrate the Jambalaya plug-in for Protégé, and to gather feedback on the tool, requests for features and new visualization ideas. The objectives of the visit were:

- to observe how Protégé is used;
- to identify areas of use requiring cognitive support;
- to gather feedback on the usefulness of the current implementation of Jambalaya; and
- to gather ideas about other potential visualizations more appropriate to the domain, and why they may be more appropriate.

The visit involved four phases:

1. an introduction to the knowledge base and its representation in Protégé,
2. contextual observations and interviews of the knowledge modeling process,
3. a demonstration of Jambalaya using a subset of the group's knowledge base, and
4. feedback from group members on their perceived utility of the visualization techniques, or on other possible or desired visualization features.

5.1 Participants: The Digital Anatomist Team

The Structural Informatics Group is an interdisciplinary team comprised of engineers, biologists and computer scientists. Two domain experts and knowledge engineers for the project participated in contextual observations. They are medical

doctors with several years experience modeling the anatomical concepts in Protégé. For the purposes of this thesis, they will be referred to as Knowledge Modellers 1 and 2 (KM1 and KM2). Other team members also provided feedback and insight into the issues they had faced creating, maintaining and sharing large knowledge bases, and the issues the users of these knowledge bases have faced as well. They include project leaders, software developers and several biologists from the team.

The next section describes the Digital Anatomist Foundational Model. Most of the information in this section comes from the project website [62], from knowledge gleaned during the visit, or from the project knowledge base itself.

5.2 The Digital Anatomist Foundational Model (FMA)

In the field of medical informatics, it has been recognized that medical practitioners rely excessively on memory and opinions to guide how they perform their professional duties [63]. Unfortunately, memories can be flawed, and learned opinions can quickly become pseudo-facts in the minds of individuals if they are not exposed to external sources of knowledge. According to Friedman *et al.* [63],

> "acting in the belief that one's personal knowledge is correct, when in fact it is wrong, is certainly one of the major causes of suboptimal professional performance in health care."

However, external "knowledge from the world" must be easy to seek out, understand and integrate with more easily accessible "knowledge in the head", like memories, to be helpful [30]. One grand goal of the field of medical informatics has always been "the creation of knowledge resources that facilitate the incorporation of 'knowledge of the world'" into professional medical practice.

The Digital Anatomist Foundational Model of Anatomy (also known as the Foundational Model of Anatomy or FMA) developed by the SIG group aims to be one such knowledge resource. This model is "concerned with the representation of concepts and relationships necessary for the symbolic modeling of the structure of the human body in a form that is understandable to humans and is also navigable by machine-based systems" [62].

Underpinning the FMA project is the belief that anatomy, the study of an organism's structure, is the foundation of all biomedical sciences. Symptoms of diseases are regarded as attributes of anatomical structures: for example, the swelling of an organ

can be regarded as an aberration of its size. Therefore, the project aims to produce a representation of human anatomy that can be reused by any biomedical applications requiring anatomical knowledge. This knowledge therefore needs to be comprehensive, unlike anatomy atlases or textbooks that tailor the knowledge represented for a specific user group. It also needs to be explicit and coherent to support machine-based inference. This is an important requirement for the development of next-generation, "smart" applications in education, clinical practice and research.

5.2.1 Implementation and Tools

Given the requirements of both human and machine understanding, the FMA is modeled as an ontology. According to the project website[62],

"the FMA is a formalism that represents a coherent body of explicit declarative knowledge about human anatomy as an ontology."

The SIG members implemented this ontology using Protégé-2000 because of its frame-based system and the support for machine inference it offers.

Containing approximately 70,000 concepts, over 100,000 terms, and over 1 million iterations of 120 types of relationships, the FMA is one of the largest computer-based knowledge sources in the biomedical sciences. This huge knowledge base is maintained in a MySQL database, where it can be accessed either through the Protégé tool, or through a web-based interface called the Foundational Model Explorer (FME). The FME was created to act as a simplified interface to the knowledge base, and it allows users to browse the knowledge base without being able to modify its contents. It is written using Java Server Pages technology, which was chosen because it can easily make use of the publicly-available Protégé Java API, and it can be used to control access to and presentation of the knowledge base – features that are either lacking in flexibility or non-existent in the Protégé tool.

5.2.2 Model Components

Four interrelated components make up the FMA: anatomy taxonomy, anatomical structural abstraction, anatomical transformation abstraction, and metaknowledge.

ANATOMY TAXONOMY (AT)

The anatomy taxonomy classifies anatomical entities according to the characteristics they share and by which they can be distinguished from one another. This distinction is represented in Protégé using the primary hierarchy: the inheritance hierarchy. In this hierarchy, the relationship between child and parent concepts can be thought of as the "is-a" relationship: for example, a lung "is-a" organ. (In fact, as shown in the screenshot of part of the Protégé taxonomy in Figure 10, the lung is differentiated from other organs in that it is solid, instead of cavitated, parenchymatous, meaning that it is composed of an organ part called the parenchyma and connective tissue, and lobular, meaning that the parenchyma is subdivided into lobules).

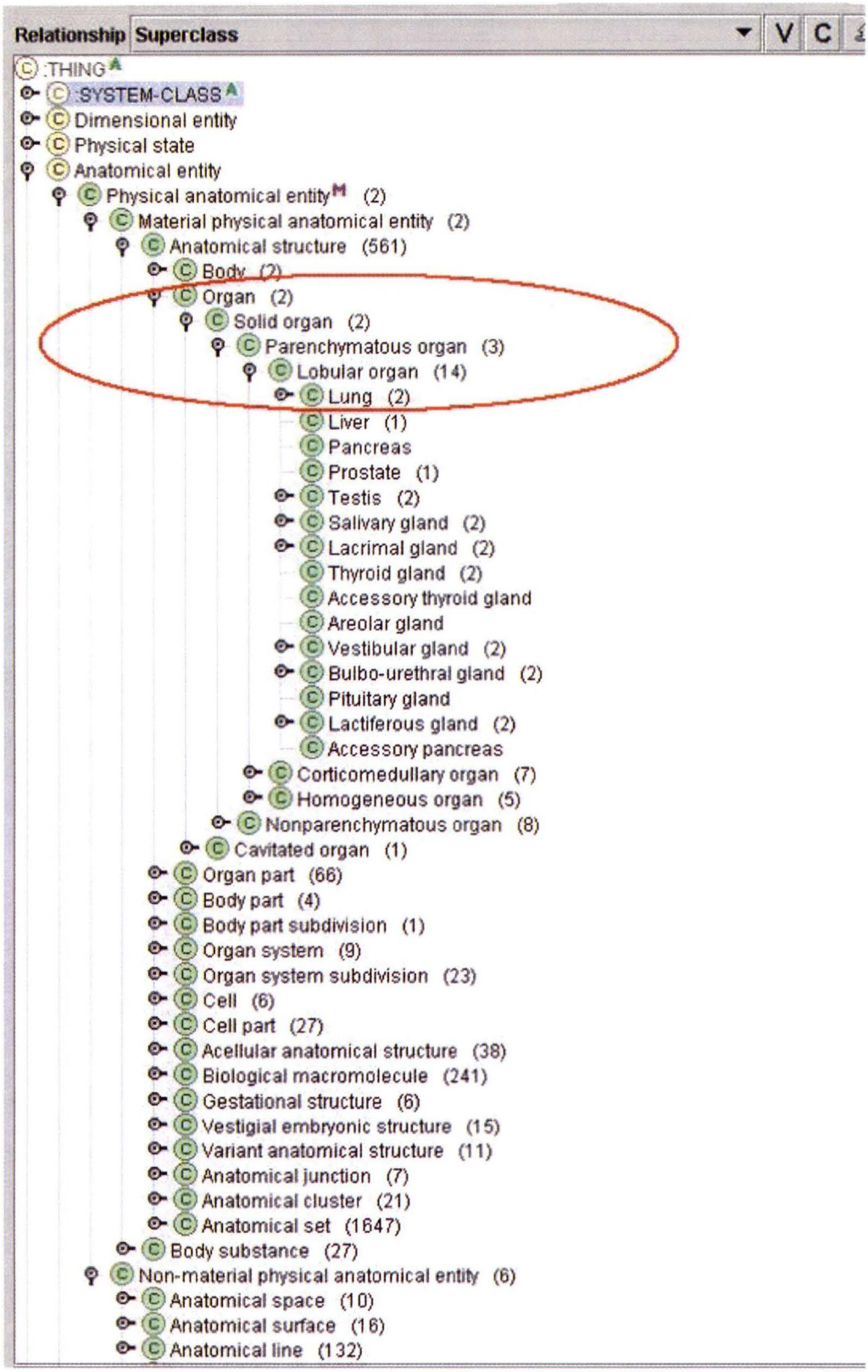

Figure 10 - The FMA inheritance hierarchy represented in Protégé. The area circled in red shows where the "Lung" class fits into this hierarchy.

The dominant concept in the taxonomy is that of the anatomical structure, defined as including "all material objects generated by the coordinated expression of groups of the organism's own structural genes." This includes biological macromolecules, cells and their parts, tissues, organs and their parts, as well as organ systems and body parts or regions. Apart from anatomical structure, the taxonomy also includes body substances (such as blood), spaces (such as the nasal cavities), surfaces, lines and points. These are all defined in terms of their relationships to anatomical structures.

ANATOMICAL STRUCTURAL ABSTRACTION (ASA)

The ASA specifies the dimensional, boundary, part-whole and spatial relationships that exist between the entities represented in the anatomy taxonomy. These relationships are modeled using slots of each concept, or frame, in the Protégé representation of the taxonomy. For example, the frame that represents the "Lung" class has a slot named "part-of", which is given the value of the class "Chest". The frame representing the "Chest" class also has a slot to represent the inverse relationship: the chest is composed of several parts, including the lung. It is important to distinguish between the "part-of" relationship and the "is-a" relationship used in the anatomy taxonomy: while the femur may be "part-of" the leg, the relationship between the two is not an "is-a" relationship. This can be confusing to people outside of the knowledge engineering domain, who are more accustomed to hierarchical decompositions of the body based on its parts.

Spatial relationships are also modeled using slots: spatial relationships defined in the model include adjacency, location, containment, and orientation, to name a few. Complications arise in the FMA because of its comprehensive nature when modeling these relationships. While most factions in the bio-medical world can agree that the heart is an organ or that the arm is a body part, the meanings of the part-whole and spatial relationships vary between user groups. For example, while the "Lung" class is related to the "Chest" class by the part-of relationship, it is also valid to say that the lung is part of the cardio-vascular system. In the FMA, this is modeled using a separate relationship, a slot named "systemic part of". The "Lung" class has the class called "Lower Respiratory Tract" as its value in this slot. Different user groups will regard these relationships with differing levels of importance: a surgeon may think of the "part-of" relationship as it is

modeled in the FMA, while a clinician may regard the "systemic-part-of" relationship as the "real" part-whole relationship of interest.

Humans easily understand the inherent similarities and differences between these two relationships: both represent a physical decomposition, one of a system, the other of an anatomical entity. To group these relationships and distinguish them, humans use their knowledge of the domain and the relationship names. A software program, however, is not able to make these inferences. With no knowledge of the meanings of the slot names, the two relationships are as different as the "part-of" relationship and the "Author" relationship, relating a concept to the person who entered its information.

The grouping and distinction of relationships in a knowledge base needs to be clear to both humans and machines. To address these conflicting needs knowledge engineers use a technique referred to as "reification" of relationships. The concept of reification refers to the treatment of an abstraction as if it were a real thing, a material entity whose character can be completely empirically decided and described. In knowledge modelling, it is used to clarify when a relationship can have more than one meaning: instead of viewing a relationship as representing an abstract and subjective concept, knowledge engineers try to decide all the possible ways in which the meaning of the relationship can differ, as if the relationship were a real object that could be measured and decided. This view allows knowledge modellers to assign attributes to relationships, as though they were not abstractions: these reified relationships are also called attributed relationships.

For example, in the FMA, the "part-of" relationship can be used to describe how an anatomical entity is part of a system, such as the lung being "part of" the cardio-vascular system. But the same relationship can be used to describe the anatomical decomposition of anatomical structures: the lung is also "part of" the thorax. The "part of" relationship is given an attribute to distinguish the possible meanings: they can be attributed as "systematic", "anatomic", or "arbitrary" part-of relationship.

Protégé does not provide a mechanism to add slots to relationships – this would be the equivalent of adding slots to slots, and slots are restricted to only having values. Instead, reification is achieved by representing relationships by instances, or frames. These frames relate the concept of interest, (for example, the lung,) the related concept,

such as the cardiovascular system, and the attribute of the relationship, such as "systemic" or "anatomic" in the case of the "part-of" relationship.

ANATOMICAL TRANSFORMATION ABSTRACTION (ATA)

This part of the model specifies the morphological transformation of the entities represented in the anatomical taxonomy during prenatal development and the postnatal life cycle; essentially, it models embryonic development. Currently no data have been entered for this part of the model.

METAKNOWLEDGE (MK)

In the FMA, metaknowledge specifies the principles, rules and definitions according to which concepts and relationships in the other three components of FMA are represented. Just as the term "metadata" specifies "data about data", metaknowledge refers to the knowledge about the concepts in a knowledge base. The FMA is an interesting knowledge base example in that it seeks to model not only anatomical constructs, but knowledge about those constructs as well. It seeks not only to model parts of the human body (such as an organ), but also what knowledge about that body part may be of relevance in the medical field (such as the normal size ranges for that organ).

The FMA has been modeled so that applications can create instances of its concepts (for example, an instance of the "Lung" class might be called "John Doe's Lung") and automatically verify that the attributes of that instance fall within the normally accepted ranges for all the attributes of interest specified in the FMA. Therefore, the anatomical concepts themselves are modeled as classes, not instances; this contrasts with most knowledge bases that contain more instances than classes. The knowledge to be modeled for each anatomical concept must be decided at a higher level, the *metaknowledge level.* Protégé supports modeling different knowledge for different classes using metaclasses; using the frame-based system, each class in the hierarchy is an instance of a metaclass.

By default, new classes created in Protégé are created as instances of the default metaclass. This metaclass specifies only that each class should have a textual name, and be allowed to have slots (or attributes). However, in the FMA, each new concept or class might have different properties of interest: the medical knowledge of relevance is not the same for every organ, for example.

Therefore, for each new concept modeled in the FMA, modellers must also model a new metaclass specifying its attributes of interest. This leads to the duplication of the hierarchy of concepts at both the class and metaclass levels. To simplify knowledge modeling, the team decided to merge the two hierarchies at a very high level (at the "Physical Anatomical Entity" branch under "Anatomical Entity"), modeling each new concept as both a class and a metaclass. Using this approach, knowledge engineers no longer have to create two frames (a class frame and a metaclass frame) for each concept in the knowledge base. Instead they can specify in the same frame the attributes of interest for a concept (such as "Name" or "Normal Size Range") and values for those attributes (such as "Lung" or "30 to 40").

5.3 Contextual Interviews and Observations

Contextual observations took place on the University of Washington campus on the morning of May 28th, 2003. Verbal permission to audiotape these sessions was obtained, and both participants signed consent forms identical to the form included in Appendix A.

In the first session, I observed KM1 as he explained the details of the FMA. He explained the purpose of the project, how the group anticipates the knowledge base will be used, and demonstrated the structure of the model and its four components. Using the Protégé interface, he demonstrated the structure of the knowledge base and concepts it contains as he explained them, as he would have to a new member of the team. This is in keeping with the "expert-apprentice" model of interaction described in [61] by Beyer and Holtzblatt[3]. He further explained the metaclass hierarchy in the model, as described in the previous section, and the reified relationships used by the group. He also demonstrated how he performs knowledge-modeling tasks in his day-to-day duties, and discussed how the team collaborates to resolve conflicts in the model.

KM1 contributed several interesting suggestions for the use of visualization in knowledge engineering. The first involved the problem their group had observed with modeling meta-data: he expressed that knowledge engineers would like to visualize not only the taxonomy hierarchy but the meta-data hierarchy as well. He also drew a rough

[3] These authors describe how the "expert-apprentice" model is an effective means of collecting data, as it creates the right behaviours on both sides of the relationship for effective learning.

suggestion for an interface for the knowledge base in which nodes nested on one relationship at a level could be nested on another relationship in the next. Further elaboration of this idea is included in the discussion section of this chapter.

Discussion with one of the project leaders between sessions revealed the particular challenges their group faces to increasing adoption of the FMA. He explained that different user groups are interested in developing applications using the FMA for their particular purposes, but are uninterested in large portions of the knowledge in the FMA – knowledge that is potentially of great use to other user groups. The comprehensive nature of the model is often overwhelming to users, and they simply want to "see" the concepts and relationships of interest to them. For example, the National Cancer Institute is interested in using the FMA, but they want only a subset of the knowledge base based on the "part-of" hierarchy. Providing interfaces that allow users to explore the FMA at a high, abstract level that is not overwhelming, while still allowing them to delve into their particular area of interest in the detail afforded by the model, has been and continues to be a challenge to the group. To increase adoption, they would also like some way to present the FMA that demonstrates what kind of information is part of the FMA, without overloading and confusing the user.

In the second session, I observed KM2 as he modeled the nerve supplies to different muscle groups – a process he called "enervating muscle groups". KM2 indicated that he had started enervating muscle groups at the bottom of the body and systematically moved up the body, indicating the nerve supply for each muscle group encountered. The afternoon of the observations, he was working on the muscles in the lumbrical of the hand.

5.4 Results

I observed several different scenarios during the contextual observation. In the first case, KM2 enervated a correctly-modeled muscle with a correctly-modeled nerve; in the second, he enervated a correctly-modeled muscle with a nerve that had not been part of the FMA; and in the third, he enervated an muscle that had been previously enervated incorrectly, with an incorrectly-modeled nerve. This combination of scenarios was interesting because it represents a spectrum of work practices: the optimal case (case 1)

when no errors are found in the knowledge base, the case when the knowledge base is found to be incomplete so new knowledge must be added (case 2), and the case when errors or inconsistencies are found and corrected (case 3). A detailed list of the actions taken by KM2 in each case can be found in Appendix B.

These three scenarios lead to the flow diagram description for the task of enervating muscle groups depicted in Figure 11.

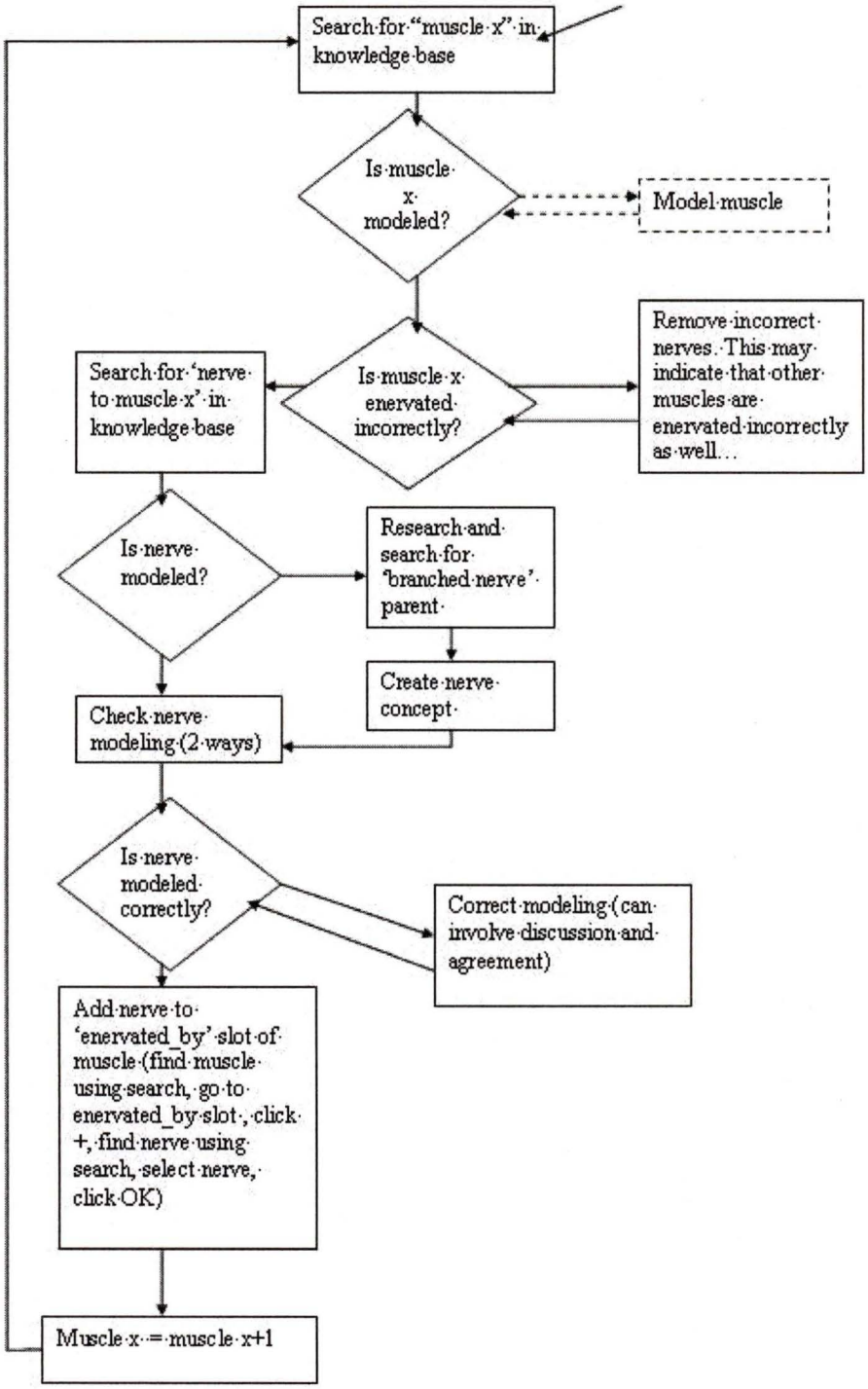

Figure 11 - A use case diagram representing the process of adding a nerve supply concept to a muscle concept. Steps shown in dashed lines were not observed directly during contextual observations sessions.

5.5 Jambalaya Demonstration and Feedback

5.5.1 Demonstrating Jambalaya

To gather feedback on our implementation to date, a demonstration of the Jambalaya plug-in was conducted for members of the SIG team, including several knowledge modellers, software developers and domain experts in anatomy. The demonstration used a subset of the FMA taken from an earlier implementation. The real FMA was not used for the demonstration as there were problems using the Jambalaya plug-in with the FMA; the knowledge base is so large that it takes much too long for the plug-in to load it[4].

During the demonstration, I described some of Jambalaya's basic features and how they might solve some of the problems I had observed or that group members had mentioned. This included demonstrating:

- hierarchies are presented in a nested graph view, the zooming controls, and how the main zooming view of the hierarchy is co-ordinated with the collapsible tree view of the hierarchy on the left of the screen
- the co-ordination of the hierarchical and thumbnail views, and how the arc and node filters are used
- how users can choose any relationship as the nesting relationship, using the "branch" relationship as an example, since I had observed it being used as a structural relationship in the user observations. Users were interested in the possibility of using the "part-of" relationship as the nesting relationship, allowing users to explore the information in a top-down manner starting at the concept of "body" down through the various "part-of" decompositions
- how flattening the graph allows users to see all the classes and instances in the knowledge base, and how the filtering mechanisms and the automatic layouts can be used to highlight interesting interactions among relationships. In particular I demonstrated how transitive relationships and reification could be shown. In Figure

[4] Subsequent changes to the Jambalaya plug-in allow users to load part of a knowledge base, based on a selected set of concepts and relationships.

12, the 'Adjacency relationship and the 'Value' relationship are displayed in blue, and the force-directed layout is applied. Not only does this view highlight the immediate adjacency relationships (as between the "T8 part of the esophagus" and the "Azygos Vein"), it shows transitive adjacencies, such as those between the "Azygos vein", the "Esophagus", and the "Trunk of thoracic duct", as shown in the left of Figure 12. This view is also useful for identifying "orphaned" instances. If a knowledge engineer creates a reified relationship between two concepts by creating an instance, they might simply remove the link between one of the concepts and the "reified instance" to remove the relationship. This can lead to the existence of reified instances that do not relate two concepts, like those in the right of Figure 12, and these should be removed from the knowledge base. The team noted that although there were mechanisms in place to automatically remove these instances when a knowledge engineer removes reified relationships, there might still be "orphaned" instances in the knowledge base from before the system was in place. Checking each instance to ensure that it is related to two concepts would be painstaking and slow, as the instance contains information only about the "related part", not the part relating to the instance.

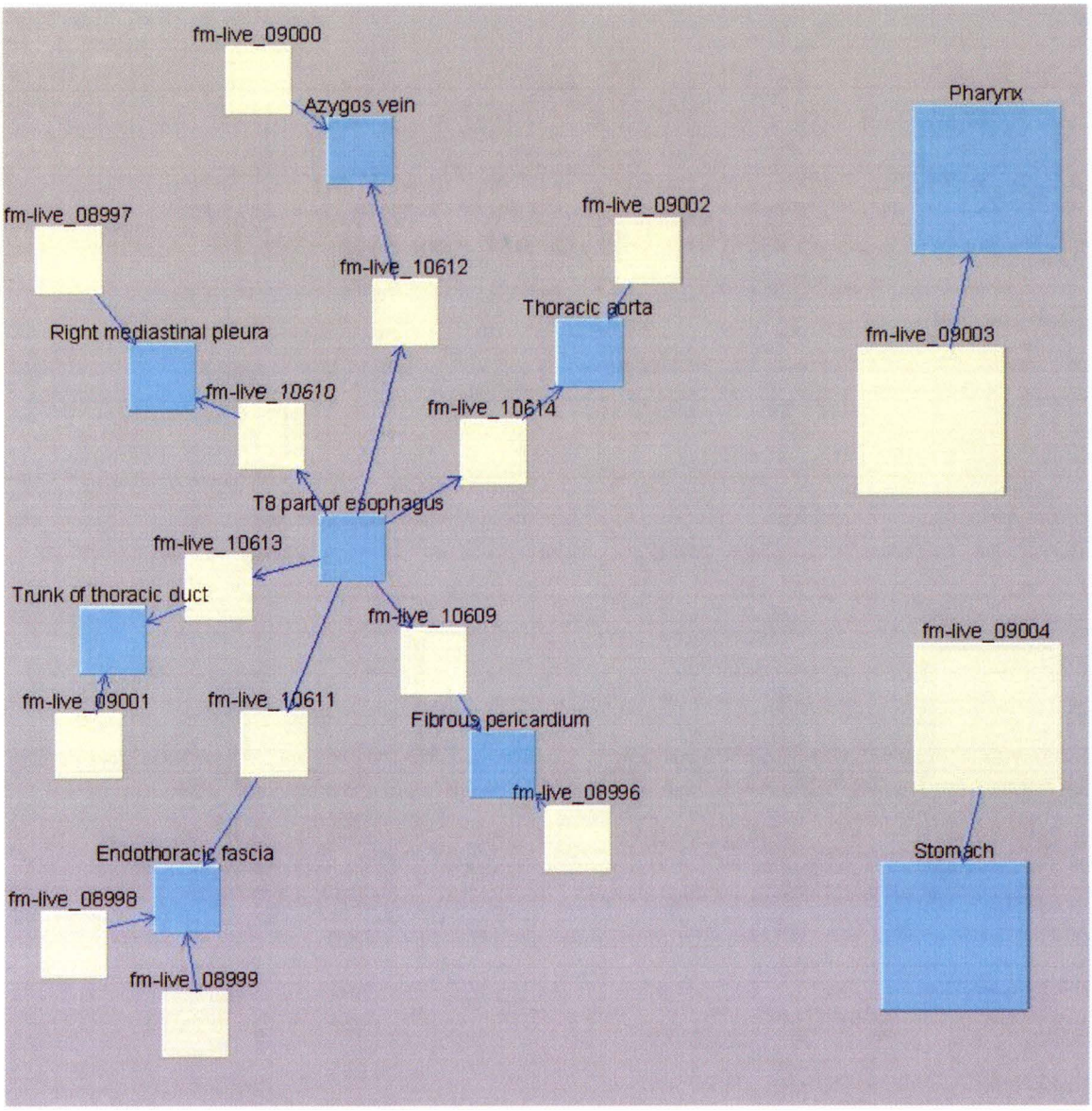

Figure 12 - The adjacency relationship from a part of the FMA displayed in Jambalaya

5.5.2 Feedback

Much of the group feedback on Jambalaya and its potential usefulness for SIG members was positive: members used words like "slick" and "very nice" to describe the interface and agreed that the zoomable nested graph interface abstracts away information but still allows users to investigate the knowledge base opportunistically on a deeper level. The knowledge engineers stated that Jambalaya could potentially be very useful for finding errors in the FMA. Currently errors are discovered and corrected on an *ad-hoc* basis as they are discovered during knowledge modeling activities.

Users also asked about other features available or potentially available in Jambalaya, such as sharing interactive views of the knowledge base with others and text searching of the knowledge base. I demonstrated the filmstrip feature for taking and sharing snapshots, and the Jambalaya search functionality.

Several concerns about the tool did arise. While the group agreed that nesting was useful for abstracting information the end user might not care about, they pointed out that nesting on a single relationship was not particularly useful. They also reacted quickly when asked about whether the zooming would be useful to knowledge engineers or if it would become tedious. I explained that the zooming provides context that allows the modellers to know where they are in the knowledge base. "They already know where they are!" said one respondent. The knowledge engineers agreed that it would be preferable to be able to jump quickly between two known areas of the knowledge base for modelling tasks. Finally, concerns arose about the scalability of the visualizations to a knowledge base the size of the FMA. Currently, all nodes not connected by a relationship are placed at the bottom of the screen when an automatic layout is applied: the team indicated that this would simply not work for a model as large as the FMA, and that these nodes should be hidden in some way.

5.5.3 Desired Visualizations

After the demonstrations, team members were asked if there were any features or different visualizations they would find useful in their work. As mentioned previously, one large barrier to the adoption of the FMA by other groups and applications is the inability to extract only a subset of the knowledge base relevant to a particular user group. The team brought forward that a tool that easily highlighted the different reified relationships separately would be very useful. They asked if it would be possible to display reified relationships between two concepts without showing the reification instance that joins them. This would involve users specifying reified relationships so that the instances could be abstracted from the visualization. The creation of different views of the knowledge base would be even more useful if these views could then be exported as stand-alone knowledge bases containing the concepts and relationships that had not

been filtered. However, this would be technically difficult because of the high degree of interconnectedness between the concepts in the FMA.

Team members also suggested a solution for knowledge modellers who frequently move between two or three areas of the knowledge base repeatedly to model a relationship between two concepts. One member used the term "recent favourites list" to describe an interface where knowledge engineers could drag and drop concepts of interest onto a list on the side of the screen. The users could then simply click on the name of the concept of interest to return to its form. This vision is slightly different from the filmstrip in that it focuses on recording the concept of interest, and not necessarily the view (which includes such things as the layout and positioning of other nodes and the current nesting relationship).This feature might also help the user in finding a concept quickly once the nesting hierarchy has changed.

Another desired feature was brought up during the feedback session that had been mentioned by KM2 during the contextual observations – the ability to map conceptual nodes to some physical representation. KM2 mentioned that while he was modeling he would sometimes lose track of a concept's physical place in the body, and that a visual reminder, some kind of photo or physical representation, would be very helpful. Other suggestions included mapping concept nodes related to different parts of the body to their related parts on an outline of a body.

A biology domain expert described his idealized interface for exploring the FMA, which he called a "star view". He wanted to start with any concept, and immediately see what properties were recorded about that concept in the knowledge base. He could then chose any of these properties to appear as arcs leading away from the node, similar to the way the "is-a" arcs are shown in the TGVizTab. He envisioned being able to select any property, and clicking on it to add the concepts related by that property to the visualization. The user could make many relationships visible in this way, and related or reified relationships would appear grouped together coming out of the concept's node. Double-clicking on a property could make the arcs and nodes for that relationship type visible in the entire tree – allowing the user to use that relationship as the structural arc for the graph of the entire knowledge base. The user could then choose to follow any of these arcs to the next concept, where they could continue to follow a relationship trail, or

they could "explode" the node into a star as well, showing all of its immediately-related concepts with arcs. The expert discussed being able to "see" both of the "exploded" nodes simultaneously, indicating that he envisioned the tool "zooming out" to show more than one node and how the nodes of interest were related.

5.6 Discussion

5.6.1 Issues in FMA Modeling Tasks

For the FMA project, the individuals modeling the knowledge involved acted as both domain experts and as knowledge engineers: while KM1 and KM2 are both medical doctors they are also both very familiar with Protégé and its frame-based representation scheme, as well as issues in knowledge modeling. Both are also very familiar with this particular model, so contextual inquiry techniques did not show these users performing many high-level model understanding tasks; both experts already have elaborate mental models of the concepts in the ontology and how they are related. Further investigation in other contexts is required to determine if these levels of understanding, both of the domain being modeled and the model and modeling tools themselves, is common for knowledge engineers in general.

NAVIGATION DURING MODELING TASKS

During the modeling tasks that were observed, it became apparent that the modellers spend a lot of time performing text searching operations, frequently searching for the same few terms. Modellers were observed typing in a search term, highlighting the term with the mouse cursor and using the Ctrl+C keyboard shortcut to copy the term to the clipboard to avoid re-typing it later. They were aware that they would be jumping between two detailed areas of model; users noted during the observations that these leaps were very common whenever they were modeling a new relationship or concept, and that the task being observed was not unusual in that regard. They also mentioned that they would consider opening two separate versions of the knowledge base to avoid repeating these navigation steps; however, using the model in Protégé required so much memory that this option is impractical.

The modellers also mentioned the importance of the 'inverse relationship' feature in Protégé to their work. A slot can be marked as an "inverse" slot in Protégé which means

that a reciprocal relationship exists: for example, the "parts" slot of the "Lower Respiratory Tract" and the "part-of" slot of the "Lung" class are inverse relationships. Once two slots have been marked as inverse relationships in Protégé, entering a value in one slot will automatically complete the value to the other slot. In the "Lung" example, if a new concept was added to the model and the value of its "part-of" slot was assigned the value of the "Lower Respiratory Tract", this new concept would also be added as a value to the "parts" slot of the "Lower Respiratory Tract" class. This feature helps modellers to add a new concept without moving to each frame to which it is related to add incoming relationships. Users were observed frequently navigating between concepts by clicking on slot values. Their level of reliance on inverse relationships indicates that they find it important for two-way navigation to be possible through this mechanism. In fact, slot-value navigation seemed to be a primary form of navigation for the modeling team.

When inverse relationships are not used, problems navigating the knowledge base arise. For example, in the reification of relationships in the FMA, inverse relationships were not used appropriately. Instances (representing attributed relationships) that related two concepts did not "show" incoming relationships: the instance relating "Lung" to "Lower Respiratory Tract" has a slot relating to "Lower Respiratory Tract" but none to "Lung". Users can navigate from the "Lung" class to this instance but not back again; there is no indication of where an incoming relationship is coming from or indeed if it exists at all.

MODEL VERIFICATION DURING MODELING

Modellers were observed checking that a concept was modeled correctly in two ways. First, they would navigate to a related concept and check that the first concept was correctly modeled in any inverse slots. Secondly, they changed the hierarchical relationship in the collapsible tree view to make sure that a new concept or relationship was being modelled correctly. Unfortunately, they again had to use text searching to find the concept they had been looking at once the hierarchy was changed.

The Digital Anatomist team is currently looking at how they can perform verification of the model they have created; currently, errors in modeling, either unintentional slips or mistakes made because of a misunderstanding of the model, are discovered by the modellers as they work. In the observations, I saw an example of a

mistake that had been made because the concept of the medial nerve had not been completely modelled. The modellers noted that errors in modeling were found about once every two weeks. Other errors are sure to exist as well: during the feedback session, team members said that while new measures were in place to prevent "orphaned" reified relationship instances, they still found them occasionally in the knowledge base.

CO-ORDINATION AND COLLABORATION ISSUES

Modellers noted that much of their decision-making about modeling decisions is done face to face, and sometimes by e-mail. Furthermore, they indicated that e-mail discussions were perceived as less effective than personal interaction. When asked if they felt that they could continue to make these decisions effectively in a much larger group or a group that is dispersed geographically, they said they didn't know how, or if, it would be possible.

5.6.2 Issues in Using the FMA

Several members of the team talked about issues in adoption of the FMA. One of the team's primary concerns is increasing the widespread use of the FMA. However, they faced barriers in expressing the content and usefulness of the FMA to different groups; this was one of the motivations for the creation of the FME, the web-based front-end. Through the FME, users can browse the knowledge base to see if it suits their purposes. However, Dr. Rosse reported that users are frequently overwhelmed by the volume of information available and the level of detail. Different user communities are only interested in certain entities, properties and relationships, and are only interested in garnering whether the FMA contains information related to those areas of interest. For example, user groups have expressed interest in using the FMA but are only interested in knowing how anatomical structures are related by the "anatomical part-of" relationship: they are interested in acquiring a small subset of the knowledge base. Other users want relationships and details abstracted away, but want to be able to investigate these relationships later in an *ad-hoc* fashion, to explore what is in the knowledge base.

Two suggested interfaces for performing this exploration arose from user discussions: a zoomable view that allows nesting on different relationships at each level, suggested by KM1 during observation sessions, and the "star" view described by one of

the participants. Future user experiments might help determine which of these approaches would be most effective and desirable for users.

5.7 Summary

The tasks performed by the expert knowledge engineers during this case study can be summarized by the following list of knowledge engineering activities:

- Adding new concepts (including relationships between concepts)
- Modifying existing concepts
- Removing concepts
- Moving concepts to another part of the ontology
- Instantiating concepts
- Editing instantiations of concepts
- Removing instantiations of concepts
- Model-domain consistency verification
- Checking that a newly-modeled concept is modeled correctly
- Model-model consistency verification
- Query-directed navigation of the knowledge base (to search for the concepts being chenged or deleted, or to find concepts related to a concept being added)
- Work assignment and division of labour
- Gaining awareness of work assignment activities

In this chapter we have identified knowledge engineering tasks and issues that will be further explored in Chapter 7. However, we first examine our own experience with knowledge engineering tasks in an introspective case study conducted within the CHISEL group in Chapter 6.

Chapter 6: Case Study

As identified in Chapter 4, we sought to expand on the results gleaned from the observational case study in Chapter 5, and to gain experience and domain knowledge in the field of knowledge engineering. This chapter describes our experiences, as they relate to knowledge engineering, designing and developing the Shrimpbib system.

6.1 Motivation

In our experience building Jambalaya, the CHISEL group members have often played the role of "novice" knowledge engineers, trying to understand new domain models given to us as examples. To gain first-hand experience as "expert" knowledge modelers, our group needed the experience of developing a knowledge base in a familiar domain. It needed to be a part of an application that would remain useful to the group so that we could become familiar with the evolving model over time.

To this end, and to better understand the tasks involved in creating and using a knowledge-based system, the author decided to implement a system that could be used by CHISEL group members, using the Protégé-2000 and Jambalaya tools. This would allow first-hand knowledge and experience of the tasks involved and the tools under examination, and would provide easy access for observation to a group of expert users – the CHISEL group members themselves. To ensure that the system would be used enough to allow for realistic feedback and evaluation in the future, the author tried to find a real problem in our group that could be solved with a knowledge-based system.

6.2 The Problem: Organizational Knowledge in an Academic Setting

Organizational knowledge refers to the shared knowledge a group has about its organization – useful lessons from previous experience, the distribution of expertise within the group, and resources most pertinent to the group's interests are all examples of organizational knowledge. Groups that manage and maintain their organizational knowledge have been found to have higher capacities for absorbing and making use of new knowledge [64].

In industry, companies have long sought to facilitate access, sharing and re-use of organizational memories using knowledge-based systems. These systems are typically based on formal ontologies, which provide explicit formal specifications of the terms in a domain and relations among them [65]. The systems are useful for formalizing and enabling the reuse of the knowledge shared by a group of people [66].

Smaller groups, such as those in academic settings, can also benefit from the capture, organization and sharing of knowledge. Academic groups, especially in university settings, tend to experience high turnover rates of personnel. Whether it is formal knowledge about particular research topics, or informal knowledge about the social network within the group, such knowledge is closely tied to the specific student, and often leaves with them; consequently, new additions to the group are forced to re-acquire this knowledge through experience.

In our group, and indeed most academic groups, research papers play a significant role in knowledge exchange. Researchers write and read papers to capture or learn more about a given discipline. Collaborating researchers will exchange papers and discuss their opinions of these papers. Connections made informally between papers can be seen in emails and web logs. Some groups may use a central repository for storing relevant research documents. Unfortunately, these repositories often lack structured and meta-information about their contents, such as who has read which papers, each reader's opinions about these papers, and how the readings fit the group's research interests.

6.3 System Goals

Although capturing all of the knowledge that is created and exchanged in our group is a challenging task, the modest goal of the Shrimpbib system is to explore whether we could at least capture knowledge about the papers of relevance to our research projects and interests. The objectives of this system are to:

(1) record organizational knowledge about the group and members' areas of participation and expertise;

(2) record group knowledge pertaining to academic documents, including members' subjective opinions of a writing's relevance to the group; and

(3) provide easy mechanisms to update and use this knowledge base with minimal disruptions to current work practices.

The Shrimpbib system is an integration of several tools. Figure 13 presents a diagram of how these tools are integrated. First, it makes use of a commercial, off-the-shelf tool called EndNote for integrating bibliographic references with popular word-processing tools. EndNote can import many existing bibliographic data formats. Though very useful when creating new documents, EndNote lacks the collaborative and meta-data support needed for this application. The Protégé tool and programming interface were used to create these features.

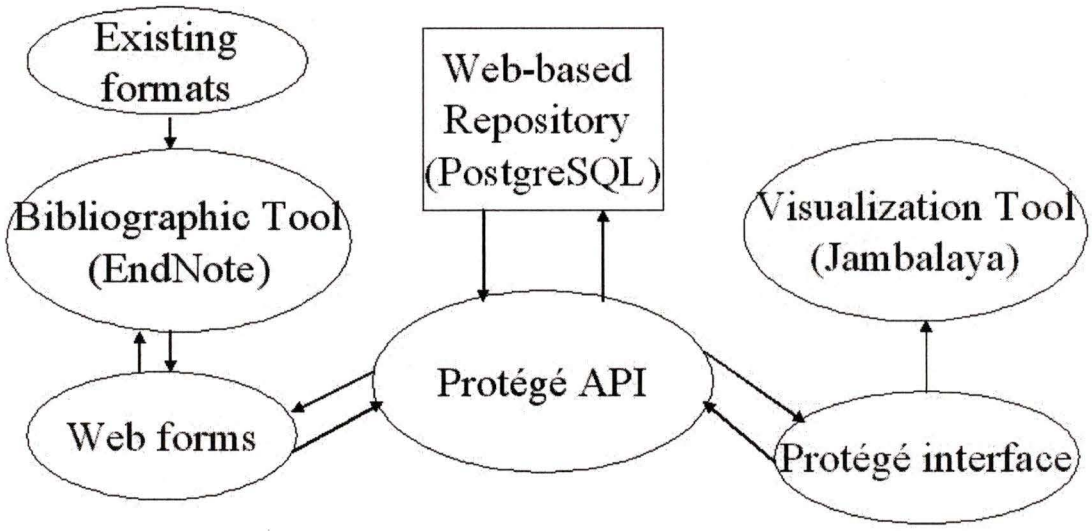

Figure 13 - A diagram of the Shrimpbib architecture of integrated tools

Although group members familiar with Protégé and knowledge modeling can enter papers and other informal knowledge directly using the Protégé interface, a simpler, lighter-weight web front-end for entering new papers and annotating existing papers was created. It is hoped that users will be more likely to use the system if there is less overhead associated with its use. While both the web interface and Protégé provide convenient mechanisms for capturing knowledge about our papers, they have limited mechanisms for exploring information and relationships that are in the knowledge base. Later in this chapter, the usefulness of Jambalaya for this task is explored.

Using this integration of different tools, a knowledge management system to capture information pertinent to the group was built. The CHISEL group members are currently using the system by populating it with references and through the visualizations we provided, encouraging further exploration of related research within our group.

6.4 The Shrimpbib System: Design, Implementation and Evaluation

This section describes the methodology followed for designing the Shrimpbib system. The method follows similar phases to those suggested by Staab *et al.* [66].

6.4.1 Phase 1: Background and Requirements Gathering

According to Staab *et al.*, the goals of this first phase are to identify and describe the end users of a potential system, and to identify opportunities for how the organization might benefit from the use of knowledge management tools. The goal, domain and scope of the ontology should be decided. Finally, applications that will be supported by the ontology should be examined. As part of this step, ontologies that have already been developed should be examined for potential re-use.

In our case the potential users are the members of the CHISEL group, as well as collaborators at other universities and companies.

Observations of our group work practices revealed most of our organizational knowledge is gleaned and shared informally, either through face-to-face discussions, weekly lab meetings, web log entries or e-mail. Previously, we had no formal mechanism for maintaining or managing this knowledge. Several members reported keeping a list of citations they intended to use, each in a different format. Most "expert" knowledge about a bibliographic citation – its relevance to the group, its content or subject matter – was not recorded electronically, if it was recorded at all. The main mechanisms for sharing papers of note with other members of the group were informal: members e-mailed papers to the group or to individuals they knew might be interested, mentioned the papers at weekly lab meetings, or posted the papers as links on their web logs.

Group introspection revealed some opportunities for leveraging information about papers and information contained in members' web logs. Our informal collaborations were already very helpful in our research tasks, where common informal queries we made of each other included:

- Which group members have expertise in a given research area?
- What are the group's areas of interest and expertise?
- What readings are recommended, either by an individual or by the group, to gain expertise in one or even several subject areas?

These queries were based on informally shared knowledge about past and current research activities. The group also acknowledged that new members would benefit from a system that could answer these queries and provide guidance about where to start reading, given access to all the readings any CHISEL member had ever cited as a reference or submitted for group consideration.

GOAL, DOMAIN AND SCOPE

The goal of the Shrimpbib ontology is to track and analyze the CHISEL group members' areas of expertise and academic readings, including projects, people, academic areas of interest, academic documents, and their bibliographic references. The domain therefore includes computer science research groups, their activities and interests. At this stage, the scope of the project was deliberately limited to support one research group. Other projects like Citeseer [67] and Dspace[68], on the other hand, are more oriented to large-scale KM projects.

SUPPORTED APPLICATIONS

To encourage the continued use and evolution of the system, several software applications need to be integrated or supported. Since paper and proposal writing makes up a large part of research activities, the integration with citation software could be of particular use to the group. Commercially available citation software stores a searchable database of academic references, and integrates with popular word-processing tools such as Microsoft Word. Users can insert citations as they write papers and theses, and citation numbers and bibliographies are automatically generated and updated by the software. CHISEL members decided on EndNote, largely because it provides sophisticated citation integration with word processors, and also because it has relatively

simple import/export functionality. EndNote, although beneficial by itself, has several limitations: it is not multi-user capable and only barely collaborative (you could, for example, share reference libraries among group members, but Endnote itself has no notion of different users); nor does it store the meta-information needed, such as the quality ratings of the papers; it has a limited query interface; and most importantly, it lacks the sophisticated facilities we require to organize the papers according to a structured terminology of research areas. Endnote accepts only free-form keyword descriptions as annotations.

While one of the goals of this case study was to gain first-hand insight by using Protégé and Jambalaya for a knowledge engineering project, there were also other reasons to use Protégé as our knowledge engineering toolkit. Protégé supports the RDF-XML serialization, which provides for potential integration with other semantic web tools and systems, as well as automated discovery of knowledge resources using RDF parsers and query tools. It can also connect to any JDBC-compliant database system to support the sharing of a common knowledge base between multiple users and locations. Protégé currently has limited multi-user support and does not allow multiple users to access the same parts of the database simultaneously. However, more support for multiple users is currently being added to the tool. Protégé is actively supported by hundreds of users worldwide in many knowledge domains, all of which contribute to its stability and ease-of-use.

Furthermore, Protégé is an open-source Java program with a comprehensive Application Programming Interface (API). This API allows it to be integrated with other Java applications and technologies. Using the Protégé API with Java Servlet technology, we were able to create a web interface for data entry and retrieval. This provides an easy way to control and track access to the knowledge base – features not currently supported in the Protégé client. A web page is also commonly accessible from any computer, without requiring users to download and install the Protégé software.

KNOWLEDGE SOURCES

Several members of the group had already organized their references using the commercial citation software package EndNote. This package also provides the ability to import bibliographies in several popular citation formats, including BibTeX for LaTeX.

EndNote citation libraries can be exported in user-defined formats, allowing us to import many citations into the Shrimpbib knowledge base. Other sources of information include user web logs, and knowledge elicited from the group about team projects and members.

POTENTIALLY REUSABLE ONTOLOGIES

As part of the Semantic Web initiative, several namespaces have been developed to describe relationships between people, such as the Friend of a Friend (FOAF) ontology [69] and to describe online documents, like the Dublin Core (DC) namespace [70]. These existing ontologies provide defined structures that we can leverage in constructing a knowledge management tool.

We performed a search for other ontologies dealing with the same topic area (computer science research and literature). While several (such as [71]) have been developed as research projects, none were comprehensive enough for our needs; we therefore recognized we would have to create much of the model anew.

We intended to develop this ontology collaboratively within the group. Group members add keywords where appropriate, and the ontology maintainer reclassifies the concepts as needed – either by the removal of duplicates, re-treeing of children, or creation of new branches as needed. To support this, there is a placeholder concept in the ontology that is named "user-added concepts". This collaborative ontology development allows our group to experience collaboration and co-ordination issues, prevalent in knowledge engineering, first-hand.

6.4.2 Phase 2: Implementation

Using Protégé, an ontology based on the FOAF and DC namespaces was created. We extended the DC 'document' concept to include several popular reference types as subclasses: conference proceedings, journal articles, magazine articles, books, and web pages. The properties for these subclasses matched fields for these document types in EndNote citation software, to facilitate a one-to-one mapping. We also extended the ontology to include ratings and annotations for each document. Each document rating was constrained to an integer between 1 (not relevant to the group) and 5 (essential reading for group members).

A screenshot of the class hierarchy in the Jambalaya tab is provided in Figure 14. This type of view was used extensively during the creation of the system to verify that newly-modeled concepts were created as expected in the knowledge base. In this screenshot, the nodes that represent the concept hierarchy of research areas is highlighted in red; note that Jambalaya highlights multiple inheritance in this hierarchy. The document hierarchy is highlighted in red. The namespaces for Dublin Core and FOAF are replicated using the prefixes ns_0_1 and ns_0_6 respectively.

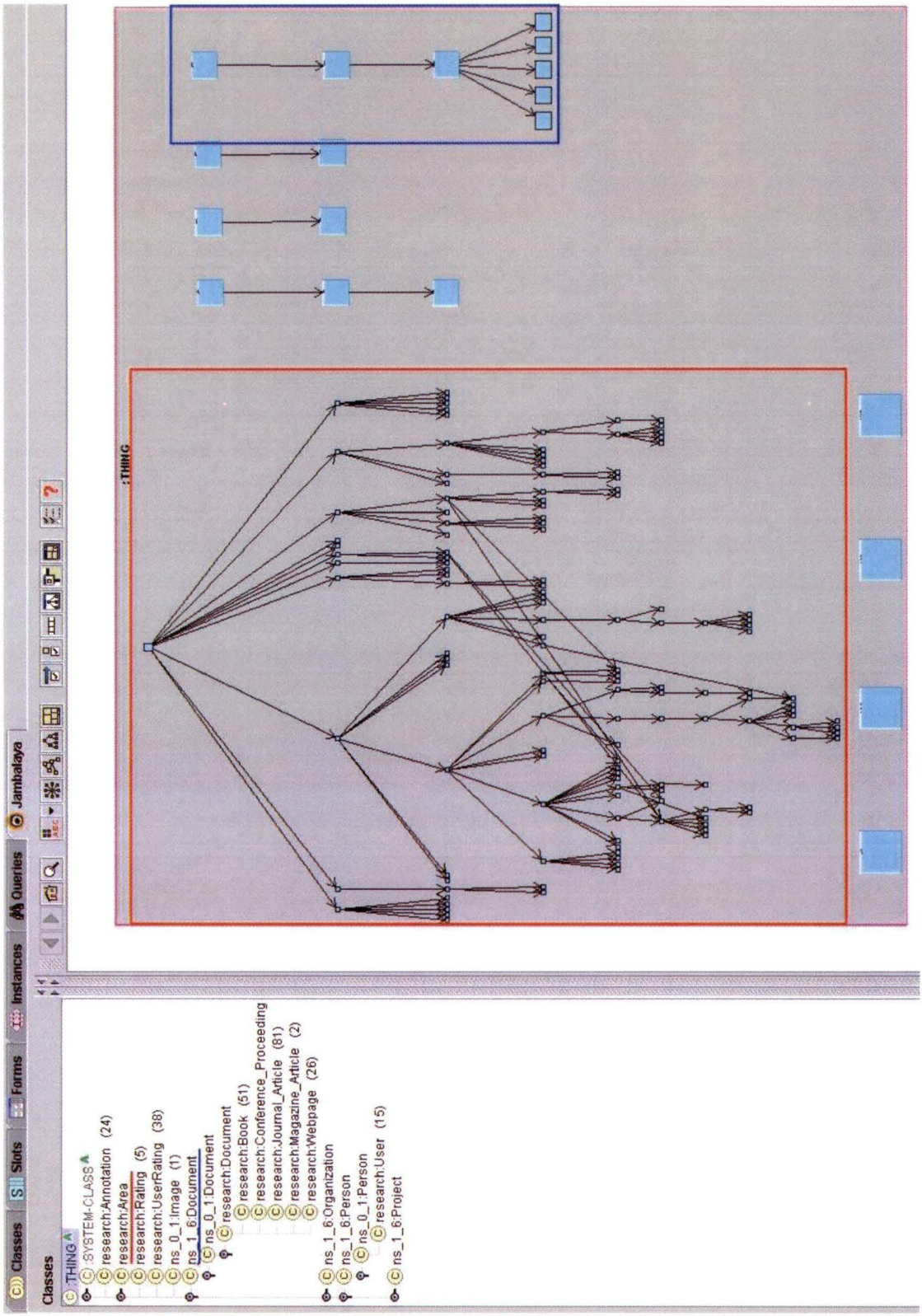

Figure 14 - A screenshot of the Shrimpbib ontology in the Jambalaya tab

Members were encouraged to start entering their references into Endnote citation libraries, or to provide their references in a format that is easily converted into Endnote, such as BibTeX. EndNote citation libraries were then exported using user-defined filters to an RDF-XML format based on our ontology. These instance files could then be merged into the Shrimpbib KB using the Protégé "import project" feature. It is worth noting that while there was a start-up cost associated with this phase, (asking users to adapt their work practices by using EndNote), users have been willing to make this effort because of the perceived benefit in using the common citation software and the benefits of being able to access the knowledge in Shrimpbib at a future time.

One drawback of using this approach to acquire information for the knowledge base was that the EndNote citation libraries did not include any rating or annotation data for each reference. As in [48] we assume that cited works could be considered as "recommended", at some level; therefore, for all visualization purposes, unrated papers were treated as if they had been given an average rating (a value of 3 on the 1-5 scale). Group members were also asked to go through the knowledge base to annotate and rate any notable papers for new group members as a 5 (essential reading).

WEB INTERFACE IMPLEMENTATION

For future reference papers, users can enter citation information, with ratings and comments, through a web interface to the shared Protégé repository. Initial efforts to build this interface using PHP, an HTML-embedded scripting language, showed that integration with Java APIs such as Protégé's is still difficult with this language. We therefore created out web application using Java Server Pages, which provided excellent integration with Protégé.

Through the web interface, users can check if a reference they have just read is already in the repository and annotate or rate it if it is; otherwise they can add the reference to the knowledge base. Users can also download all the references in the knowledge base in a generic format that is easily imported into EndNote. Individual members can thus use the entire volume of group reference data for citations when writing papers or theses.

6.4.3 Phase 3: Evaluation phase

In this phase, developers need to assess if the system can answer the knowledge-related questions identified by its users. Therefore, we assess if we can answer the user questions we identified in Phase 1.

QUERY 1: WHICH GROUP MEMBERS HAVE EXPERTISE IN A GIVEN RESEARCH AREA?

In Fig. 15, nodes representing documents and group members are connected by blue arcs representing the "User Documents" relationship, which associates documents with the users who have added or annotated a given document. "Document" nodes are further redundantly coloured based on the group member that added the document to the knowledge base. When one applies a force-directed layout algorithm [72], visual grouping of the nodes easily shows how many papers each user has added.

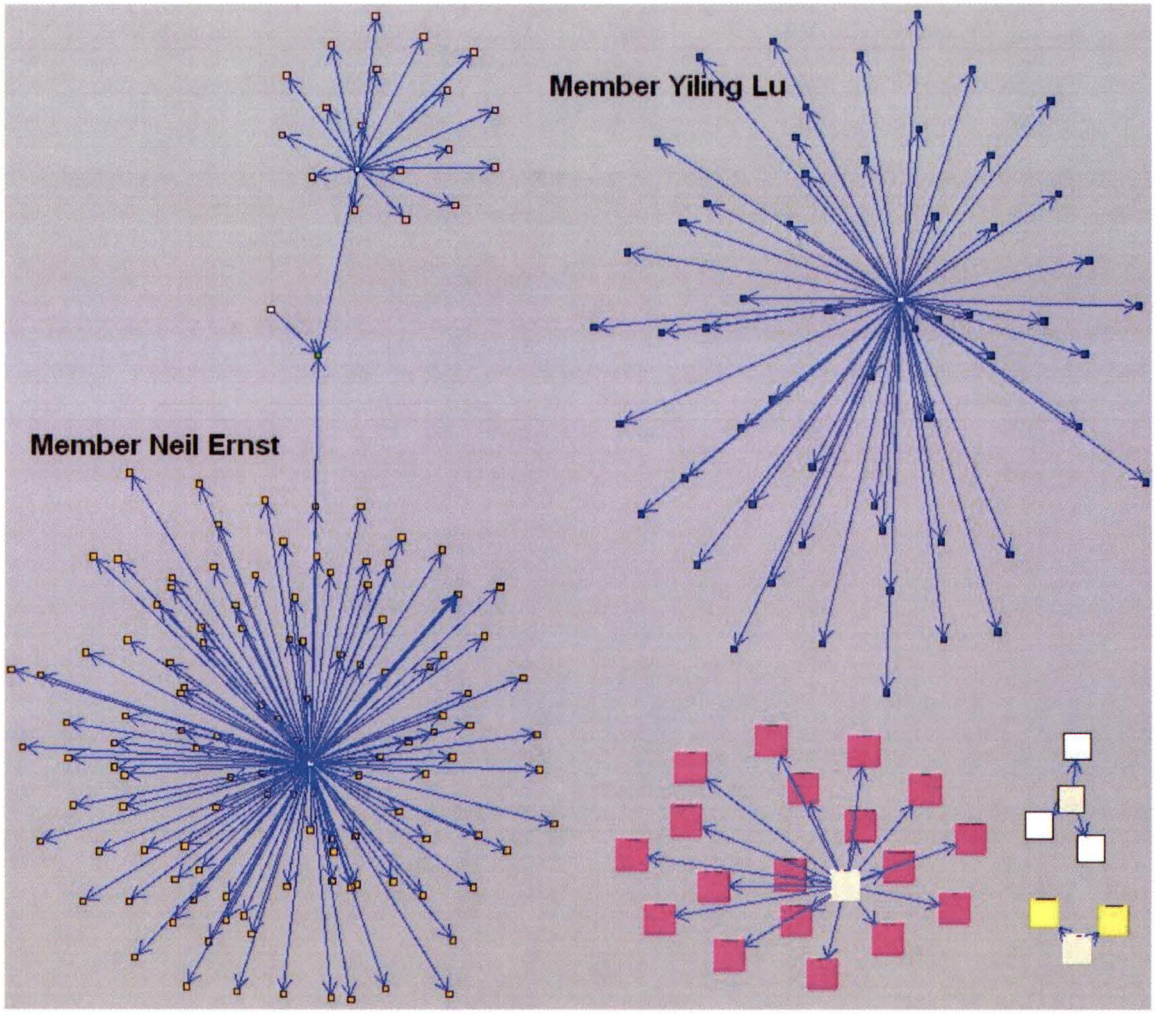

Figure 15 - A view of Shrimpbib showing group members and their associated documents

In Fig. 16, we add another relationship of interest, the research areas associated with each document (shown in red). Nodes representing documents, group members, and research areas are connected by these two relationships, and the same layout algorithm positions nodes that have more inter-connections between them closer together on the screen. Using this visualization, users can discover which group members have expertise in various research areas. For example, we can see that researchers "Yiling Lu" and "Neil Ernst" have submitted many papers related to information visualization, while only researcher "Neil Ernst" has submitted papers relating to artificial intelligence.

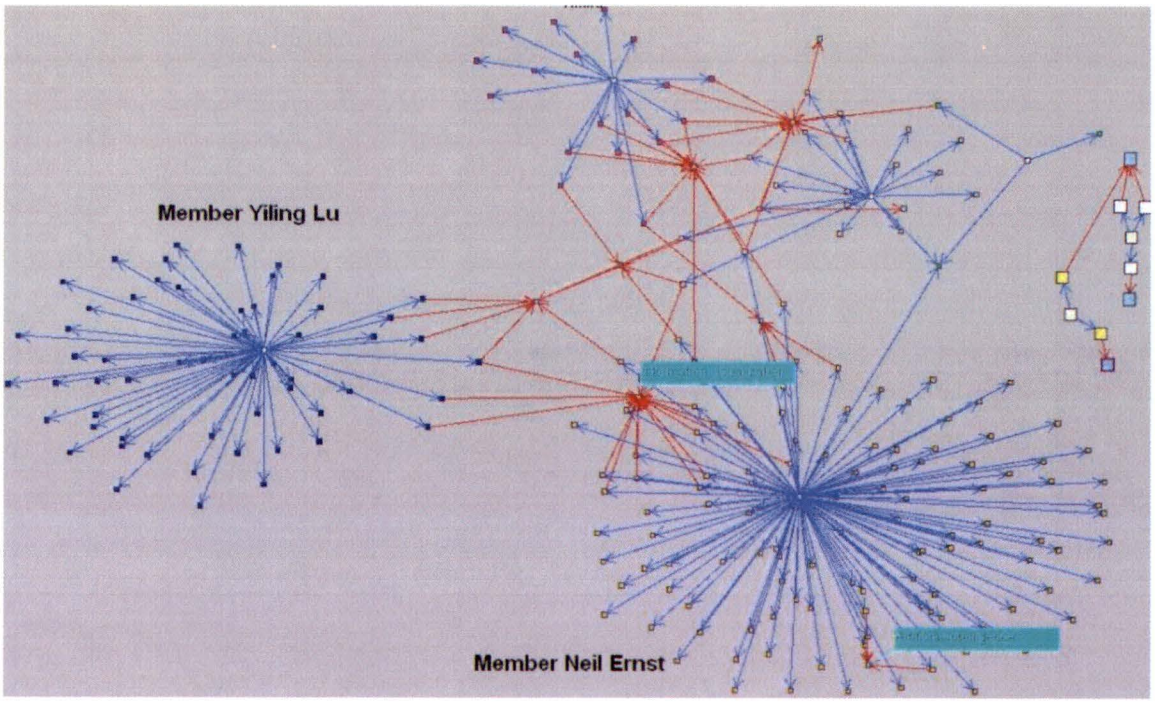

Figure 16 - Another view of Shrimpbib showing group members, documents and areas of interest. The blue arcs relate group members to the documents they have added, while the red arcs relate these documents to their related area.

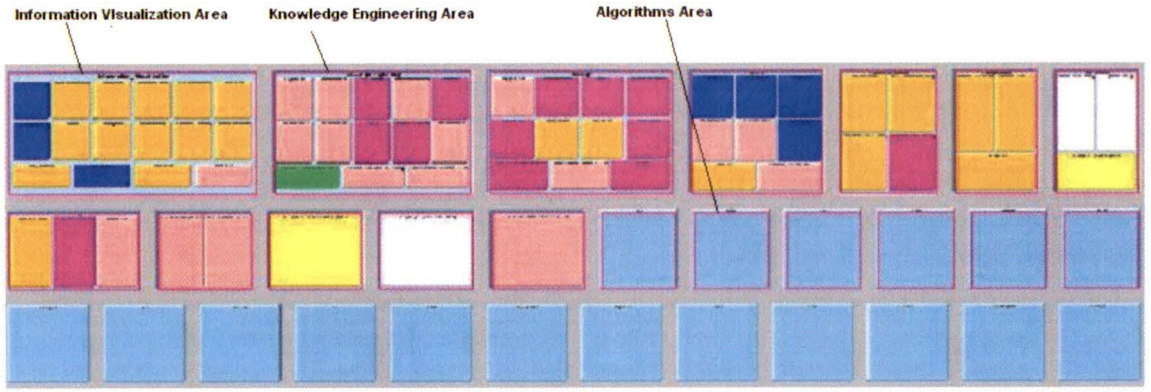

Figure 17 – Nodes representing documents are nested within nodes representing areas of interest, and colored based on the user that added the document.

QUERY 2: WHAT ARE THE GROUP'S AREAS OF INTEREST AND EXPERTISE?

A possible visualization for exploring the group's areas of expertise is presented in Fig. 17. In this representation, each document node is nested inside nodes representing areas of interest. A treemap layout [73] re-sizes each 'area of interest' node based on the number of documents the area contains. Users can quickly see, based on the size of each area node, which areas have many related documents. For example, the group has read many documents relating to information visualization and knowledge engineering, but none relating to algorithms.

Since document nodes are again assigned a colour based on who submitted the document, this visualization can also help users determine which group members have submitted documents related to each area.

QUERY 3: WHAT READINGS ARE RECOMMENDED, EITHER BY AN INDIVIDUAL OR BY THE GROUP, TO GAIN EXPERTISE IN ONE OR EVEN SEVERAL SUBJECT AREAS?

In Fig. 18, each document node is nested inside a node representing one of the five possible ratings. This view allows users to identify the works considered essential reading, either by the group (by looking at all documents rated a "5") or by a particular group member. Alternatively, documents could be coloured by research area to show recommended papers in each area. Either view provides essential guidance to new group members about background reading.

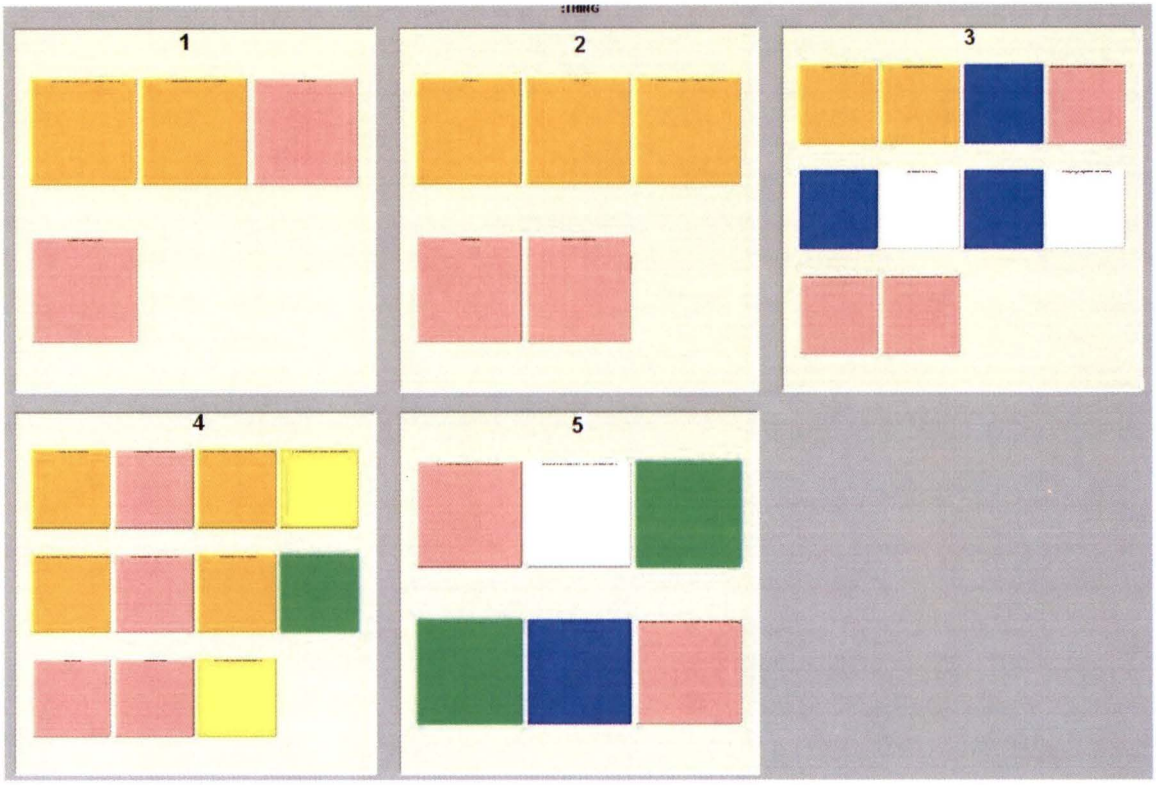

Figure 18 - Document nodes are nested inside nodes representing their associated rating

Initial reaction to the system from the target user-base has been favourable. After seeing demonstrations of the visualizations presented here, group members immediately grasped the power of the tool and its ability to show previously hidden relationships. They also appreciated the usefulness of the integration of this knowledge base with citation software, and several group members went to the effort of creating EndNote libraries of their bibliographic references specifically so they could be included in the knowledge base.

6.4.4 Phase 4: Evolution and Maintenance phase

Even without changing the ontology that underlies the Shrimpbib application, effort is required to maintain the knowledge base. Currently, users may accidentally add instances of documents that are already represented in the knowledge base; this was particularly problematic when importing instances from multiple EndNote citation libraries. Even for a small group, the effort required to merge duplicate instances was not insignificant. Since a user interface is not currently provided to end users for correcting errors, this maintenance effort also falls to the creators of the system.

Parts of the Shrimpbib ontology are still being refined. In particular, the "research area" ontology will probably continue to change significantly for the near future, although its structure is expected to evolve more gradually over the long-term. Maintainers must continue to look through new user-created area terms, held in the "user-added concept" placeholder, on a weekly or even daily basis, in order to classify these new areas or merge them with existing ones.

Based on user feedback about the system, the Shrimpbib tool continues to be refined. Current efforts continue to increase the integration with the EndNote software, so that transferring information between EndNote and Shrimpbib will be easy and will not disrupt work practices. Eventually we hope that the Shrimpbib system can act as a user interface to a digital repository of documents read by the group that would be invaluable to all group members.

6.5 Summary

The tasks observed during the obserational study described in Chapter 5 were all repeated at some stage during the development of the Shrimpbib system. However, though our first-hand experience, some further tasks important to the knowledge engineering process also emerged:

- Knowledge base design: re-using and combining existing ontologies and namespaces
- Verifying that the model includes necessary concepts and relationships
- Exploration, "opportunistic" investigation of the knowledge base
- Understanding the structure of the knowledge base
- Understanding the content of the knowledge base
- Sharing/publishing the structure of a knowledge base
- Sharing/publishing the content of a knowledge base

The experience of the knowledge engineering process that we gained by building a knowledge-based system and using it within our group has lead to some important insights relevant to the needs of knowledge engineers in general. In Chapter 7 we will examine the results obtained in Chapters 5 and 6 to determine their implications.

Chapter 7: Synthesis of Results

Based on the experience of the contextual observations of the SIG team at the University of Washington and the case study conducted at the University of Victoria, several categories of tasks relevant to knowledge engineering have emerged:

1. **Knowledge Modeling Tasks**

   These include tasks performed by the knowledge modelers and domain experts to design ontologies, add knowledge to a knowledge base and to maintain the knowledge base structure and contents.

2. **Model Verification Tasks**

   During knowledge modeling, knowledge engineers were observed verifying that newly modeled concepts were modeled appropriately. However, these verification tasks could also occur separately from knowledge modeling tasks.

3. **Model Usage Tasks**

   End users of the system wish to re-use the knowledge acquired and modeled. They need to explore and query the knowledge base to achieve understanding at various levels – either at the structural level, to determine what kind of knowledge the system contains, or at the content level, to understand the domain itself.

4. **Co-ordination and Collaboration Tasks**

   Knowledge bases are rarely created by a single author. Therefore their creation, maintenance and usage require collaboration between knowledge engineers, domain experts and end users to iteratively model concepts appropriately.

During user observations and throughout the development of the Shrimpbib system, many issues arose that highlighted the need for tools that better support both the knowledge engineers attempting to design, implement and maintain the system, and the domain experts attempting to contribute knowledge and use the final product.

The next sections discuss these tasks and their implications for information visualization tools in knowledge engineering (including suggested changes to Jambalaya), and then describe how they can be used to guide future, less exploratory evaluations of these tools.

7.1 Implications for Knowledge Engineering Tools and Visualization Support

7.1.1 Knowledge Modeling Tasks

These tasks include knowledge base design, adding and describing new classes and instances to the knowledge base, and adding relationships between these concepts.

KNOWLEDGE BASE DESIGN

During the creation of the Shrimpbib ontology, we found little support for re-using or combining parts of existing ontologies and namespaces. We wanted to add in existing schemas such as Dublin Core and FOAF without creating separate Protégé projects, perhaps by directly referencing an external universal resource indicator (URI), thus allowing the external developer/maintainer to update the ontology as appropriate.

KNOWLEDGE BASE CREATION

For experienced modellers, a common navigation mechanism during authoring tasks involved extensive use of the Protégé text searching. To help users repeating search terms many times, a simple solution could involve adding a drop down list of recent terms to the Protégé search dialog. However, further support could be offered for what seems to be a prevalent task during authoring: working on two separate but very focused parts of the tree. As suggested by comments from the SIG group members (such as "knowledge engineers already know where they are") and by previous user studies at IBM involving business flow diagrams, such authoring tasks might not require the context support offered by a zoomable interface and might instead benefit from a screen-splitting, or multi-pane approach. Users could then 'jump' easily between two or more focused areas of interest.

7.1.2 Model Verification Tasks

Both the SIG group observations and the Shrimpbib case study revealed two types of model verification: checking a newly-modeled concept is modeled correctly, and browsing the knowledge base for errors (in either domain or model consistency).

VERIFYING NEWLY-MODELED CONCEPTS

SIG modellers were observed changing the primary hierarchy in Protégé to verify that a newly added concept was modeled correctly; they then had to search for the added concept in the new hierarchy. This reflects a need to verify that the added concept is modeled correctly in multiple hierarchies simultaneously, and this task may benefit from visualization support such as that provided by Jambalaya. Using Jambalaya, multiple relationships can be shown using arcs allowing for easier verification. Modellers in the Shrimpbib system had this functionality available; they never saw the need to change the primary Protégé hierarchy.

BROWSING FOR ERRORS

While browsing the knowledge base, SIG members relied heavily on navigation through slot values: by clicking on the value associated to a concept, that value's form would appear in a new pane. The group members noted that the 'inverse slot' functionality in Protégé was invaluable to their work – indicating that a two-way path of slot values was important to their navigation of the knowledge base. However, in many cases, inverse relationships had not been created. One potential solution would be for the Protégé tool to automatically create inverse relationships for all slots; however, for large knowledge bases this option is impractical because of memory requirements. Another alternative is available through visualization: in Jambalaya, a concept's incoming relationships are visible along with outgoing relationships. In the case of the FMA, this type of visualization could be particularly useful in the identification of 'orphaned' instances.

Jambalaya was also used extensively in the Shrimpbib case study for the visual checking of structure. This visual checking for structural patterns and anomalies is easier than looking for the same information though forms because it takes of advantage human perceptual abilities.

7.1.3 Model Usage and Understanding Tasks

In our experience in this project, we found that interactive visualization techniques show great promise as a means to support users attempting to understand the structure and content of a knowledge base. Future work and empirical evaluation is needed to determine the most useful and expressive representations and interactive features to be

implemented in Jambalaya. For example, empirical experimentation could help to determine the relative merits of the two interfaces proposed for knowledge base exploration, the "nested query" view that allows nesting on different hierarchies at different levels, and the "star view" that allows incremental exploration.

The results suggest several tasks for which users require support, such as understanding reified relationships. Understanding the interactions amongst other simple relationships is also important. Frame-based systems only support binary relationships between concepts. This means that humans must take their understanding of how some entities are inter-related, which quite possibly includes n-ary relationships, and model those relationships as multiple binary relationships. As an example, consider the "document_areas" and " user_documents" relationships in the Shrimpbib system. The first shows how academic documents are related to research areas; the second relates group members to documents they have added. As humans we understand that the three concepts (documents, research areas, and members) are actually interrelated. Visualization can take these binary relationships and represent them in a way that more closely matches our understanding of them than how they need to be represented in a frame-based system. Further investigation is needed into whether Jambalaya's approach, using arc and node filters and automatic layouts, is effective or can be improved.

For future visualization tools, new concepts are being defined in syntaxes for semantic web such as OWL (Ontology Language for the Web). This new area of growth will require support in new ways that have yet to be investigated.

7.1.5 Co-ordination and Collaboration Tasks

Both studies in this thesis involved small groups of modellers who were also experts in the domain being modeled. To support larger, more dispersed teams, a closer look at how these kinds of teams make decisions in fields like Computer-Supported Collaborative Work and Global Software Development is merited. Better ways to represent knowledge and modeling alternatives, and better ways to share those representations, are needed.

During the Shrimpbib case study it was apparent that the tools we used provide little support for the collaborative nature of ontology creation. While Protégé allows multiple

knowledge engineers to access the same knowledge base, simultaneous access using the Protégé software by multiple users caused problems. Currently, there is no support for specifying different access levels for an ontology: while domain experts should be allowed to add new knowledge, knowledge engineers may wish to specify which parts of the structured research areas vocabulary experts are allowed to modify. The Shrimpbib application does not allow domain experts to modify the ontology through the web interface - they can only add instance information to the knowledge base. The other alternative was to allow domain experts full access to modifying the ontology. However, this approach leaves no room for consensus building within a group, and could have disastrous effects should unfamiliar users (new group members, for example) modify the knowledge base inappropriately. The domain experts can suggest new concepts for the structured terminology, but we place the burden of modelling these new concepts (i.e., deciding where in the tree they should be placed) on the administrators of the ontology. The administrators must make informed decisions (possibly via consultation with domain experts) about how they should be classified. Finally, the process for importing references, even if they were specified in an RDF-XML format that matched the RDF Schema we used in Protégé, was awkward. This process was essential for building a collaborative knowledge base.

7.1.4 Summary of Implications

In this section, we have identified several areas where knowledge engineering tasks could be better supported through visualization, including:

- support for "jumping" between two concepts as they are being joined by a relationship during knowledge modeling;
- support for understanding a concept's place in multiple hierarchies simultaneously, to help verify newly-modeled concepts;
- support for the visual checking of a knowledge base's structure;
- support for the identification of a concept's incoming relationships, or the lack thereof, during model verification;
- support for incremental navigation of a knowledge base, possibly using the "star view" proposed by users at the University of Washington;

- support for opportunistic investigation into deeper levels of a hierarchy while abstracting uninteresting information, possibly through the "nested query" view proposed by the same group; and
- support for human understanding of reified relationships and n-ary relationships.

Furthermore, we identify other areas in knowledge engineering that require further support, though not necessarily visualization support, such as the reuse of ontologies and namespaces during knowledge base design, or collaboration and co-ordination tasks inherent in the building of knowledge bases.

7.2 Implications for Empirical Evaluation in Knowledge Engineering

The research in this thesis has begun to explore the tasks performed by users involved in designing, creating and using knowledge bases. Further research into these tasks at finer levels of detail than those prescribed in KE methodologies is required to determine the requirements these users have for cognitive support. The experience gained using a contextual inquiry case study and an introspective case study indicates that while these case studies may reveal important insights and overcomes the challenge of amassing enough expert users, the generalizability of such results to other knowledge bases or to other domains remains a problem. By conducting two case studies in very different contexts, and comparing and contrasting the results, the problem is slightly mitigated. Hopefully further investigations of this nature will eventually allow for a meta-analysis that could guide the development of specific requirements for information visualization tools for the knowledge engineering field.

7.2.1 Proposed User Experiment Design

Based on the tasks we observed, it is now possible to propose several possibilities for experimental designs for knowledge engineering tools.

SCENARIOS

Several approaches could be beneficial. Users who are familiar with knowledge engineering concepts could be introduced to a new domain, such as that presented in the "wines" knowledge base, which is distributed with the Protégé-2000 tool as a standard

example. This knowledge base describes the domain of wines, wineries, regions and food accompaniments for wines, and being a relatively familiar domain, would not require much user training to be understood. A comparison of results between users accomplishing tasks with and without the Jambalaya tool could show how the tool supports "novice" knowledge modelers.

"Expert" modelers would have to be more familiar with a domain; analogous tasks could be chosen from a domain in which the user group members are already experts. For example, members of the CHISEL group could use the knowledge base created and used in the Shrimpbib case study. It would have to be noted that the users' familiarity with the Jambalaya tool might be a confounding factor.

TASKS

Figure 19 illustrates how the tasks identified above relate to possible experimental tasks using the "wines" knowledge base, the example introduced in Chapter 2. Various knowledge engineering tools are devised to support different tasks in knowledge engineering; researchers seeking to evaluate those tools would therefore only select the tasks relevant to the tool under investigation. Experimental designs could involve asking users to complete these tasks in different interfaces and comparing the results, or simply asking the users to accomplish tasks with a particular tool to determine if the tool supports the tasks as intended.

This task list does not include all tasks possibly supported by knowledge engineering tools; it only addresses the tasks we encountered during observations or over the course of the introspective case study. Other tasks exist and further investigation is required into potential visualization support for them. For example, we can speculate that visualization support can aid informal design techniques during the design of an ontology; however, ontology design was not observed during contextual observations, and the Shrimpbib case study was too small an example to reveal examples of these activities.

Figure 19 - Mapping of Knowledge Engineering Tasks to the Corresponding Experimental Tasks

Knowledge Engineering task	**Corresponding experimental task in 'Wines' ontology**
MODEL DEVELOPMENT TASKS	
Adding new concepts	Add a new type of Rosé wine called "White Chianti" to the knowledge base.
Modifying existing concepts	Add a slot to this "White Chianti" class.
Removing concepts	Remove the concept of a "White Merlot" wine from the knowledge base.
Moving concepts to another part of the ontology	Modify the knowledge base so that all "White Zinfandel" wines are classified as white wines instead of rosé wines.
Instantiating concepts	Add the following wine to the knowledge base: a dry reisling white wine, from Quail's Gate Vineyard (Quail's Gate Dry Reisling) made with a Reisling grape.
Editing instantiations of concepts	Change the "Chateau Morgon Beaujolais" wine, a red Beaujolais wine, to have a "sweet" sugar level instead of a "dry" sugar level.
Removing instantiations of concepts	Remove the "Chatean Morgon Beajolais" wine from the knowledge base.
VERIFICATION TASKS	
Model-domain consistency verification	Is this information modelled as you would expect? How would you change how it is modeled? (Searching for planted errors)

Knowledge Engineering task	**Corresponding experimental task in 'Wines' ontology**
Checking that a newly-modeled concept is modeled correctly	How would you verify that the Quail's Gate Dry Reisling you added is modelled correctly in the knowledge base?
Verifying that the model includes necessary concepts and relationships	Searching for planted errors, such as missing relationships between parts of the ontology
Model-model consistency verification	Has anyone made errors entering information into this knowledge base? (Searching for planted errors)
MODEL USAGE TASKS	
Exploration, "opportunistic" investigation	Imagine the following scenario: you have just arrived at a new job, and have been told that you will be asked to answer questions based on this knowledge base. Please take a few minutes to explore the knowledge base.

Knowledge Engineering task	**Corresponding experimental task in 'Wines' ontology**
Understanding the structure of the knowledge base	Questions based on the structure, such as questions about reified relationships, and interactions among relationships. Examples: How complete is this knowledge base? Which parts are more complete than others? What kind of questions can you answer based on this knowledge base?
Understanding the content of the knowledge base	Questions based on the knowledge base contents. Examples" Is the "Cotturri Zinfandel"a red wine, or a rosé wine? What wines go best with lamb, according to the knowledge base? What are the best wineries for this kind of wine?
Query-directed navigation	Search for the "Chablis" concept (chosen because it is deeply nested and the name provides less cues as to its place in the hierarchy)
CO-ORDINATION AND COLLABORATION TASKS	
Sharing/publishing the structure of a knowledge base	Further investigation required to determine appropriate experimental tasks. The examination in this thesis did not provide many insights into these tasks.

Knowledge Engineering task	**Corresponding experimental task in 'Wines' ontology**
Sharing/publishing the content of a knowledge base	Further investigation required (as above)
Work Assignment	Further investigation required (as above)
Awareness of work assignment activities	Further investigation required (as above)

MEASURES

To determine if Jambalaya improves human performance in the tasks it is intended to support, we need to determine what measures should be taken in a user experiment. It is very hard to 'measure' human understanding beyond testing a person's ability to answer questions based on the knowledge base. However, we can measure the time taken to respond to questions about the knowledge base structure and content, and the accuracy of these responses. For verification tasks, a model with planted errors is useful so we can measure error detection and recovery times.

HYPOTHESES

Based on the theories we have built about information visualization support for knowledge engineering tools, namely the categorization of knowledge engineering tasks and the areas for potential visualization support presented in Section 7.1, a few hypotheses can be drawn about the cognitive support offered by Jambalaya. These hypotheses can be investigated further, or verified in experimental conditions, depending on the level of their maturity. For each hypothesis, the maturity of the question investigated is rated as high (ready for user experimentation), medium, or low (further theory development is required).

- We hypothesize that users will be able to detect errors in model-domain consistency and in model-model consistency more quickly and accurately using the Jambalaya plug-in than using the Protégé interface alone. (High)
- We hypothesize that users will be able to check that a newly-modeled concept is modeled correctly in fewer steps and more quickly using Jambalaya than when using the Protégé interface alone. (High)

- We hypothesize that the use of Jambalaya leads more quickly to a more complete understanding of a knowledge base's content and structure that the Protégé interface for new knowledge engineers. (Medium-High)
- We hypothesize that Jambalaya better supports opportunistic investigation of a knowledge base than the Protégé interface. (Medium)
- We hypothesize that, while the Protégé interface largely supports model development tasks effectively, the users will be able to better evaluate the effects of design decisions on an ontology if they are able to visualize the knowledge base and the possible results of a design decision. (Medium)
- We hypothesize that visualization can play a role in supporting co-ordination and collaboration tasks. (Low)

Chapter 8: Conclusion

To many researchers, the term 'empirical evaluation' is equivalent to user studies and experiments. However, these approaches to evaluation may be premature in emerging fields where theoretical foundations have not yet been established. In this investigation we found that, despite the lack of theory from which to draw clear hypotheses, evaluation using data-driven approaches gave us valuable insights into the emerging field of information visualization in knowledge engineering. These insights continue to guide the evolution of Jambalaya, the development of theories about visualization support in this field, and future evaluations based on these theories.

By beginning with explorations of the fields of knowledge engineering and information visualization, including previous evaluation techniques used in these fields, we laid the groundwork for an analysis of the maturity of these investigations. While more experimental studies of advanced information visualization techniques are beginning to be conducted, empirical work in the knowledge engineering field is just beginning to reach less exploratory stages. The use of visualization techniques for knowledge engineering tools is a nascent idea; evaluations of these techniques are correspondingly preliminary, and we concluded that further investigations into the users and tasks involved in this domain were required before useful experimental hypotheses could be made.

To gather information about the users and the tasks they perform, we turned to the field of requirements engineering. After surveying the techniques used in this field to elicit requirements, we selected two techniques based on our limited access to expert users of knowledge engineering systems, and on the techniques' appropriateness to the exploratory nature of the investigation. First, contextual inquiry and observation techniques were used to glean insights and feedback from knowledge engineers and modellers at the University of Washington Structural Informatics Group. Second, we conducted an introspective case study investigating the creation and use of a knowledge-based system for our academic group.

Given the results from these two case studies, we analysed the tasks we observed to derive a preliminary taxonomy of tasks supported by knowledge engineering tools,

including knowledge modeling tasks, model verification tasks, model usage and understanding tasks, and tasks relating to co-ordination and collaboration. We furthermore suggested several areas where these tasks could benefit from better support, including visualization support.

Finally, we use this analysis to derive suggestions for possible experimental evaluations of Jambalaya or any visualization tool for knowledge engineering, and include the hypotheses we have developed about the support offered by the Jambalaya tool to be verified or refined in later studies.

The results of this thesis are important given the increasing attention being paid to the W3C's Semantic Web intiative and to knowledge engineering in general. As more user-centered tools, such as visualization tools, are created to support knowledge base development, meaningful ways to assess these tools will become more and more critical.

8.1 Summary of Contributions

The contributions this thesis provides include:

1) A descriptive taxonomy of tasks performed by users of frame-based systems such as Protégé, including tasks involved in knowledge modeling, model verification, model usage and understanding, and co-ordination and collaboration.
2) Suggestions for cognitive support for some of these tasks, including visualization support, as described in the previous section.
3) Suggestions for future evaluations of information visualization tools for knowledge engineering, and experimental hypotheses about the cognitive support Jambalaya offers.
4) The Shrimpbib tool, created as part of the introspective case study in Chapter 6. This tool continues to provide new group members with a starting point in exploring the CHISEL group's areas of research through relevant academic literature, and efforts to increase adoption within the group continue.

8.2 Future Work

The results of this thesis have in some cases already affected the development of the Jambalaya tool. Currently, support has been added for the human understanding of

reified relationships, and prototypes of both the "star view" and "nested query" interfaces are being implemented.

Work in the CHISEL group continues towards the development of the requirements for information visualization tools for knowledge engineering. This includes investigations into the collaborative tasks in knowledge engineering and visualizations supporting them; visualization tools for comparing and merging ontologies and knowledge bases; and different possible representations for exploring and understanding knowledge bases and their comparative benefits. Many of the questions asked in this thesis are being examined and the evolution of Jambalaya, including theory-building in this domain and the verification of the hypotheses derived, will continue.

Bibliography

1. BASILI, V.R., The Experimental Paradigm in Software Engineering. in *Dagstuhl-Workshop*, (Lecture Notes in Computer Science 706, Experimental Software Engineering Issues: Critical Assessment and Future Directives, September 1992).
2. SHAW, M., Writing Good Software Engineering Papers. in *Int. Conf. on Software Engineering (ICSE)*, (Portland, OR, 2003).
3. ALMSTRUM, V.L., PETRE, M., BERGLUND, A., DALE, N., GRANGER, M., LITTLE, J.C., MILLER, D.M., SCHRAGGER, P. and SPRINGSTEEL, F. Evaluation: Turning technology from toy to tool. Report of the Working Group on Evaluation *SIGCSE Bulletin*, 1996.
4. STOREY, M.-A. and MULLER, H., Manipulating and Documenting Software Structures using SHriMP Views. in *1995 International Conference on Software Maintenance (ICSM95)*, (Opio (Nice), France, 1995), 275-284.
5. GUARINO, N. and GIARETTA, P. Ontologies and Knowledge Bases: Towards a Terminological Clarification. in Mars, N. ed. *Towards Very Large Knowledge Bases: Knowledge Building and Knowledge Sharing*, IOS Press, Amsterdam, 1995, pp 25-32.
6. STUDER, R., BENJAMINS, V.R. and FENSEL, D. Knowledge Engineering: Principles and Methods. *Data Knowledge Engineering, 25* (1-2). 161-197. 1998.
7. NOY, N.F. and MCGUINESS, D.L. Ontology Development 101: A Guide to Creating Your First Ontology, Stanford University, 2001.
8. GROSSO, W.E., ERIKSSON, H., FERGERSON, R.W., GENNARI, J.H., TU, S.W. and MUSEN., M.A. Knowledge Modeling at the Millennium (The Design and Evolution of Protege-2000), Stanford University, 1999.
9. SOWA, J.F. *Knowledge Representation: Logical, Philosophical, and Computational Foundations*, 2000.
10. SCHREIBER, A.T. and BIRMINGHAM, W.P. Editorial: the Sisyphus-VT initiative. *International Journal of Human-Computer Studies, 44* (3-4). 275-280. 1996.
11. BERNERS-LEE, T., HENDLER, J. and LASSILA, O. The Semantic Web *Scientific American*, May 2001.
12. SHADBOLT, N., O'HARA, K. and CROW, L. The experimental evaluation of knowledge acquisition techniques and methods: history, problems and new directions. *International Journal of Human-Computer Studies, 51* (4). 729-755. 1999.
13. COHEN, P.R. *Empirical Methods for Artificial Intelligence*. MIT Press, Boston, 1995.
14. NOY, N.F., GROSSO, W. and MUSEN, M.A., Knowledge Interfaces for Domain Experts: An Empirical Evaluation of Protege-2000. in *Knowledge Acquisition Workshop (KAW 99)*, (Banff, Alberta, 1999).
15. TALLIS, M., KIM, J. and GILL, Y., User Studies of Knowledge Acquisition Tools: Methodology and Lessons Learned. in *Knowledge Acquisition Workshop (KAW99)*, (Banff, Alberta, 1999).

16. Knowledge Acquisition Initiative of the Knowledge Acquisition Community (KA)2, http://www.swi.psy.uva.nl/usr/richard/ka2/research-topic.html
17. SCHREIBER, G., WIELINGA, B., HOOG, R.D., AKKERMANS, H. and VELDE, W.V.D. CommonKADS: A comprehensive methodology for KBS development. *IEEE Expert*, *9* (6). 28-37. 1994.
18. ANGELE, J., FENSEL, D. and STUDER, R. Developing Knowledge-Based Systems with MIKE. *Journal of Automated Software Engineering*, *5* (4). 389-418. 1998.
19. DOMINGUE, J., MOTTA, E. and WATT, S. The Emerging {VITAL} Workbench. *Knowledge Acquisition, Modeling and Management*. 320-339. 1993.
20. SWARTOUT, W. and GIL, Y. EXPECT: A User-Centered Environment for the Development and Adaptation of Knowledge-Based Planning Aids. in Tate, A. ed. *Advanced Planning Technology: Technological Achievements of the ARPA/Rome Laboratory Planning Initiative*, AAAI Press, Menlo Park, Calif., 1996.
21. MENZIES, T., hQkb - The High Quality Knowledge Base Initiative. in *Knowledge Acquisiton Workshop (KAW99)*, (Banff, Alberta, 1999).
22. COHEN, P., SCHRAG, R., JONES, E., PEASE, A., LIN, A., STARR, B., GUNNING, D. and BURKE, M. The DARPA High-Performace Knowledge Bases Project *AI Magazine*, 1998, 24-49.
23. MENZIES, T. Knowledge Elicitation: the State of the Art. *Handbook of Software Engineering and Knowledge Engineering*, *2*. 607-628. 2000.
24. GOMEZ-PEREZ, A. From Knowledge Based Systems to Knowledge Sharing Technology: Evaluation and Assessment, Knowledge Systems Laboratory, 1994.
25. VEGA, J.C., GOMEZ-PEREZ, A., TELLO, A.L. and PINTO, H.S.A.N.P., (ONTO)2 Agent: An ontology-based WWW broker to select ontologies. in *13th European Conference on Artificial Intelligence (ECAI 98)*, (Brighton, England, 1998).
26. JASPER, R. and USCHOLD, M., A Framework for Understanding and Classifying Ontology Applications. in *Knowledge Acquisition Workshop (KAW99)*, (Banff, Alberta, 1999).
27. MUSEN, M.A., FERGERSON, R.W., GROSSO, W.E., NOY, N.F., CRUBEZY, M. and GENNARI, J.H., Component-Based Support for Building Knowledge-Acquisition Systems. in *Conference on Intelligent Information Processing (IIP 2000) of the International Federation for Information Processing World Computer Congress (WCC 2000)*, (Beijing, 2000).
28. DUINEVELD, A.J., STOTER, R., WEIDEN, M.R., KENEPA, B. and BENJAMINS, V.R., Wondertools? A comparative study of ontological engineering tools. in *Knowledge Acquisition Workshop*, (Banff, Alta., 1999).
29. NOY, N.F. and MUSEN, M.A., PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment. in *Seventeenth National Conference on Artificial Intelligence (AAAI-2000)*, (Austin, TX, 2000).
30. NORMAN, D.A. *The Design of Everyday Things*. Basic Books (Perseus), New York, 1988.
31. MUNZNER, T. Guest Editor's Introduction to Special Issue on Information Visualization. *Computer Graphics and Applications, IEEE Computer Society Press*, *22* (1). 20-21. 2002.

32. CARD, S.K., MACKINLAY, J.D. and SHNEIDERMAN, B. (eds.). *Readings in Information Visualization, Using Vision to Think*. Morgan Kaufmann, 2000.
33. CARD, S.K. and PIROLLI, P., The Cost-of-Knowledge Characteristic Function: Display Evaluation for Direct-Walk Dynamic Information Visualizations. in *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI '94)*, (Boston, MA, 1994), 238--244.
34. FURNAS, G.W., Effective view navigation. in *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'97)*, (1997), Addison-Wesley, 367--374.
35. TWEEDIE, L., Characterizing Interactive Externalizations. in *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'97)*, (1997), 375-382.
36. PIROLLI, P. and RAO, R., Table Lens as a tool for making sense of data. in *Workshop on Advanced Visual Interfaces(AVI-96)*, (Gubbio, Italy, 1996).
37. *Special Issue: Empirical Evaluation in Information Visualization. International Journal of Human-Computer Studies*, *53* (5). 2000.
38. CHEN, C. and CZERWINSKI, M. Empirical evaluation of information visualizations: An introduction. *International Journal of Human-Computer Studies*, *53*. 631-635. 2000.
39. CHEN, C. and YU, Y. Empirical studies in information visualization: a meta-analysis. *International Journal of Human-Computer Studies*, *53* (5). 851-866. 2000.
40. FAIRCHILD, K., POLTROK, S. and FURNAS, G. Semnet: Three dimensional graphic representations of large knowledge bases. in Guindon, R. ed. *Cognitive Science and its Applications for Human-Computer Interaction*, Lawrence Erlbaum, 1988.
41. AT&T Labs Research - Graphviz Website, http://www.research.att.com/sw/tools/graphviz/
42. TouchGraph, http://www.touchgraph.com/tech_overview.html
43. Ontoviz, Michael Sintek, Stanford University, http://protege.stanford.edu/plugins/ontoviz/ontoviz.html
44. IsaViz: A Visual Authoring Tool for RDF, http://www.w3.org/2001/11/IsaViz/
45. TGViztab, http://www.ecs.soton.ac.uk/~ha/TGVizTab/TGVizTab.htm
46. KAON - The Karlsruhe Ontology and Semantic Web Tool Suite, http://kaon.semanticweb.org/
47. BORNER, K., CHEN, C. and BOYACK, K., Visualizing Knowledge Domains. in *ARIST 37*, (2002).
48. CHEN, C. and PAUL, R.J. Visualizing a Knowledge Domain's Intellectual Structure *IEEE Computer*, 2001, 65-71.
49. NG, G.K.-C. Interactive Visualization Techniques for Ontology Development *Faculty of Science and Engineering*, University of Manchester, Manchester, U.K., 2000, 273.
50. STONE, M.C., FISHKIN, K. and BIER, E.A., The Movable Filter as a User Interface Tool. in *ACM Conference on Human Factors in Computing Systems (CHI '94)*, (Boston, MA, 1994), ACM, 306-312.
51. NIELSEN, J. *Usability Engineering*. Academic Press, San Diego, CA, 1993.

52. RAYSIDE, D., LITOIU, M., STOREY, M.-A. and BEST, C., Integrating SHriMP with the IBM WebSphere Studio Workbench. in *CASCON'2001*, (Toronto, 2001).
53. GOMEZ-PEREZ, A. Some Ideas and Examples to Evaluate Ontologies, Knowledge Systems Laboratory, 1994.
54. GIBOIN, A., GANDON, F., CORBY, O. and DIENG, R., Assessment of Ontology-based Tools: Systemizing the Scenario Approach. in *Proceedings of EON2002: Evaluation of Ontology-based Tools Workshop at the 13th International Conference on Knowledge Engineering and Knowledge Management EKAW 2002*, (Siguenza (Spain), 2002), pages 63-73.
55. GAUSE, D.C. and WEINBERG, G.M. *Exploring Requirements: Quality Before Design*. Dorset House Publishing Co., New York, NY, 1989.
56. NUSEIBEH, B. and EASTERBROOK, S., Requirements engineering: a roadmap. in *ICSE - Future of SE Track*, (2000), 35-46.
57. MACAULAY, L.A. *Requirements Engineering*. Springer-Verlag, London, 1996.
58. BOEHM, B. A Spiral Model of Software Development and Enhancement *Computer*, May 1988, pp.61-72.
59. GOGUEN, J. and LINDE, C., Techniques for requirements elicitation. in *IEEE International Symposium on Requirements Engineering*, (San Diego, CA, 1993), IEEE, 152-164.
60. KOTONYA, G. and SOMERVILLE, I. *Requirements Engineering Processes and Techniques*. John Wiley & Sons Inc., New York, NY, 1998.
61. BEYER, H. and HOLTZBLATT, K. *Contextual Design: Defining Customer-Centered Systems*. Morgan Kaufmann Publishers, San Francisco, 1998.
62. Digital Anatomist Foundational Model Website, http://sig.biostr.washington.edu/projects/fm/AboutFM.html
63. FRIEDMAN, C.P., OZBOLT, J.G. and MASYS, D.E. Towards a New Culture for Biomedical Informatics: Report of the 2001 ACMI Symposium. *8*. 519-526. 2001.
64. COHEN, W.M. and LEVINTHAL, D. Absorptive capacity: A new perspective on learning and innovation. *Administrative Science Quarterly*, *35* (1). 128. 1990.
65. Ontology: different ways of representing the same concept. *Special issue of Communications of the ACM*, *45* (2). February 2002.
66. STAAB, S., STUDER, R., SCHNURR, H.-P. and SURE, Y. Knowledge Processes and Ontologies. *Jan/Feb 2001*. 26-34. 2001.
67. Citeseer Website, www.citeseer.com
68. The DSpace project. *www.dspace.org*.
69. FOAF: the 'friend of a friend' vocabulary, http://xmlns.com/foaf/0.1/
70. Dublin Core Metadata Initiative, http://dublincore.org/
71. Sure, Y. 2001. The Semantic Web Research Community Ontology, Published December 11, 2001 at http://ontobroker.semanticweb.org/ontos/swrc.html#casestudies
72. FRUCHTERMAN, T.M.J. and REINGOLD, E.M. Graph drawing by force-directed placement. *Software- Practice and Experience*, *21*. 1129-1164. 1991.
73. SCHNEIDERMAN, B. and WATTENBERG, M., Ordered Treemap Layouts. in *IEEE Symposium on Information Visualization*, (San Diego, CA, 2001), 73.

Appendix A: Example consent form

Empirical Evaluation of a Visualization for a Knowledge Engineering Tool

You are being invited to participate in a study entitled "Empirical Evaluation of a Visualization for a Knowledge Engineering Tool" that is being conducted by Polly Allen, Neil Ernst, David Perrin and Jonathan Heron. Polly Allen, Neil Ernst and David Perrin are graduate students, and Jonathan Heron is a visiting researcher in the Department of Computer Science at the University of Victoria. You may contact them if you have further questions by telephone at (250) 472-5171.

As graduate students, we are required to conduct research as part of the requirements for a degree in Computer Science. It is being conducted under the supervision of Dr. Margaret-Anne Storey. You may contact our supervisor at (250) 721-8796 and mstorey@cs.uvic.ca.

This research is being funded by the National Cancer Institute. The purpose of this research project is to perform an objective empirical evaluation of the SHriMP plug-in for the Protégé-II knowledge management tool. We also aim to determine which parts of the tool are useful for different user groups. Research of this type is important because very little empirical evaluation exists in the fields of knowledge engineering and information visualization.

If you agree to voluntarily participate in this research, your participation will include participant observation, individual and group interviews and a questionnaire. There are no known or anticipated risks to you by participating in this research. The potential benefits of your participation in this research include contributing to the state of knowledge.

Your participation in this research must be completely voluntary. If you do decide to participate, you may withdraw at any time without any consequences or any explanation. If you do withdraw from the study your data will be excluded from the study.

In terms of protecting your anonymity, no identifying information will be collected from participants.

Your confidentiality and the confidentiality of the data will be protected by keeping all paper files in a locked cabinet and maintaining all electronic files in secure, password-protected directories. Data from this study will be disposed of after three years.

It is anticipated that the results of this study will be shared with others in the following ways: directly to participants, theses, dissertations, class presentations, presentations at scholarly meetings, published articles, and/or over the Internet.

In addition to being able to contact the researcher [and, if applicable, the supervisor] at the above phone numbers, you may verify the ethical approval of this study, or raise any concerns you might have, by contacting the Associate Vice-President, Research at the University of Victoria (250-472-4362).

Your signature below indicates that you understand the above conditions of participation in this study and that you have had the opportunity to have your questions answered by the researchers.

_____	_____	_____
Name of Participant	*Signature*	*Date*

A copy of this consent will be left with you, and a copy will be taken by the researcher.

Appendix B: A detailed list of actions performed by knowledge modelers during contextual observations

CASE 1: Enervating correctly modeled muscle with correctly modeled nerve

- Finds muscle he is enervating using the Protégé text search window
- Checks in the Protégé form that no nerve is assigned to the "Enervated by" slot
- Searches for a nerve named "nerve to muscle X" using the Protégé text search window. KM2 explained he uses his own domain knowledge to recall if a 'named' nerve exists for a particular muscle group. Some nerves have been given proper names in the medical community, but many have not. The project team has established naming conventions for unnamed concepts: the established naming convention is that nerves are given the name 'nerve to x' where x is the name of the muscle enervated.
- In this case, the class "Nerve to muscle X" exists in the FMA.
- Checks that this nerve is modeled correctly:
    - Searches for the nerve from which the "Nerve to muscle X" branches off. Again, KM2 used domain knowledge of anatomy to determine which nerve to search for.
    - Checks the "branches" slot for this nerve to see if the nerve in question is modeled correctly
    - Also changes the hierarchy in the Protégé tree view to 'branch of' and checks that the "Nerve to muscle x" is a child node of the nerve it branches from
- Searches for "muscle x" using the Protégé text search
- Navigates to the 'enervated_by' slot, clicks the '+' (add) icon
- The hierarchy of all valid classes for this slot value is displayed: in this case, the slot has been constrained to subclasses of the "Nerve" class.
- Uses the text search dialog to find the "Nerve to muscle X" class
- Selects the nerve, clicks on "Ok"

CASE 2: Enervating correctly modeled muscle with non- modeled nerve

- Finds muscle he is enervating using the Protégé tree hierarchy view displaying the is-a relationship hierarchy. KM2 explains the tree view helps him to keep track of which muscles he has already "enervated".
- Checks in the Protégé form that no nerve is assigned to the "Enervated by" slot
- Searches for a nerve named "nerve to muscle X" using the Protégé text search window
- The "Nerve to muscle X" class does not exist in FMA.
- Research is required: KM2 describes this part of the task as tedious. Several different textbooks are consulted as medical texts often conflict or become outdated; often sources of information or conflicts between sources are also recorded in the FMA.
- Looks for the nerve in question in the medical textbooks to make sure it has no alternative name that might already be modeled in the FMA.
- Finds which nerve the "Nerve to muscle X" branches off of.
- Uses the Protégé text search for that parent nerve, navigates to the "branches" slot, and click on the 'C' (create) icon
- Creates a new frame for a class called 'Nerve to muscle x'.
- Checks that this nerve is modeled correctly:
    - Searches for the nerve from which the "Nerve to muscle X" branches off.
    - Checks the "branches" slot for this nerve to see if the nerve in question is modeled correctly
    - Also changes the hierarchy in the Protégé tree view to 'branch of' and checks that the "Nerve to muscle x" is a child node of the nerve it branches from
- Searches for "muscle x" using the Protégé text search
- Navigates to the 'enervated_by' slot, clicks the '+' (add) icon
- The hierarchy of all valid classes for this slot value is displayed: in this case, the slot has been constrained to subclasses of the "Nerve" class.
- Uses the text search dialog to find the "Nerve to muscle X" class
- Selects the nerve, clicks on "Ok"

CASE 3: Enervating incorrectly enervated muscle with incorrectly modeled nerve

- Finds muscle he is enervating using the Protégé tree hierarchy view displaying the is-a relationship hierarchy
- Checks in the Protégé form that no nerve is assigned to the "Enervated by" slot. In this case, a nerve has already been assigned, a class called "Medial nerve". KM2 states that this is incorrect: the nerve that enervates this muscle is a branch of the medial nerve.
- Browses through other nearby muscles – they also have the "Medial Nerve" class as a value in the "Enervated by" slot. KM2 knows this is incorrect.
- Searches for medial nerve (using text searching) to add branches and make sure at least the medial nerve is modeled correctly.
- Searches through reference textbooks and finds that the medial nerve branches off a nerve plexus; this is not modeled in the FMA knowledge base. In fact, the medial nerve is not modeled as a "branch" of any other nerve: there is no value in the "branch_of" slot in the medial nerve frame.
- Performs a text search for the nerve plexus in question
- Navigates to the "branch" slot for the nerve plexus and clicks on the '+' (add) icon
- The hierarchy of possible values for the "branch" slot, all the subclasses of the "Nerve" class is displayed. KM2 performs a text search for the "Medial Nerve" class.
- There are no results for the search for "Medial Nerve" in the nerve hierarchy.
- Performs a text search for "Medial Nerve" to find the class in the "is-a" hierarchy again.
- This time, KM2 notices that the "Medial Nerve: class is modeled as an organ. In the FMA, an organ cannot be a branch of another nerve or nerve plexus.
- Discussion with KM1 reveals that the term 'medial nerve' is ambiguous in medical terminology: it can be used to refer to either the whole nerve (from spine onwards), or just to the part of the medial nerve from the nerve plexus onwards. To formalize this, they will have to divide the 'medial nerve' organ into separate

concepts: a pre-plexus part, a plexus part, and a post-plexus part of the medial nerve. The "Nerve to muscle x" class will branch from the post-plexus part of the medial nerve.

- Division of the medial nerve concept is left for another time. KM2 will continue enervating muscle groups not affected by this change in the model.

Vita

Surname: Allen Given Names: Mary Margaret

Place of Birth: Vancouver, British Columbia, Canada

Educational Institutions Attended:
University of Victoria 1995 to 2000 and 2001-2003

Degrees Awarded:
B. Sc. (with Distinction) University of Victoria 2000

Honours and Awards:
University of Victoria Computer Science/Math Co-op Work Term Report Writing Award (September 2000)
NSERC Undergraduate Research Award (May-August 2000)
University of Victoria Award for Academic Excellence in Computer Science (December 1997)
Royal Bank Academic All Canadian Honour Roll (April 1996)

Publications:

P.M. Allen, "Requirements Engineering in Global Software Development: A Case Study", University of Victoria Technical Report

D. Damian, J. Chisan, P.M. Allen and B. Corrie, "Requirements Meets Awareness: Awareness Needs in GSD", in proceedings of the 2nd International Workshop on Global Software Development (co-located with ICSE 2003), Portland, OR, May 2003.

N.G. Leveson, P.M. Allen and M.-A. D Storey, "The Analysis of a Friendly-Fire Accident Using Control-Based Model of Accidents", in proceedings of the 20th International System Safety Conference, Denver, CO, August 2002

UNIVERSITY OF VICTORIA PARTIAL COPYRIGHT LICENSE

I hereby grant the right to lend my thesis (or dissertation) to users of the University of Victoria Library, and to make single copies only for such users or in response to a request from the Library of any other university, or similar institution, on its behalf or for one of its users. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by me or by a member of the University designated by me. It is understood that copying or publication of this thesis for financial gain by the University of Victoria shall not be allowed without my written permission.

Title of Thesis/Dissertation:

Empirical Evaluation of a Visualization Tool for Knowledge Engineering

Author [redacted]

Mary Margaret Allen

September 30, 2003