An Investigation into Knowledge Acquisition
and its Emergent Effects on Knowledge Base Quality

by

Adam Doan

A Thesis
Presented to
The University of Guelph

In partial fulfilment of requirements
for the degree of
Master of Science
in
Computer Science

Guelph, Ontario, Canada

© Adam Doan, March, 2012
ABSTRACT

AN INVESTIGATION INTO KNOWLEDGE ACQUISITION AND ITS EMERGENT EFFECTS ON KNOWLEDGE BASE QUALITY

Adam Doan
University of Guelph, 2012

Advisor: Dr. Mark Wineberg

This project presents an investigation into the viability of alternative knowledge acquisition strategies in knowledge management systems. The goal of this project is to illustrate that alternative means of knowledge acquisition can have a significant effect on the quality of the knowledge base. To accomplish this, a modification of a wiki system, dubbed Prometheus, is proposed that uses a threshold based user vote acquisition mechanism.

A simulation approach is used to compare a model of the Prometheus system against a model of a standard wiki system. A simulation framework is described that facilitates comparison between models of knowledge systems. The simulation framework is used to compare the knowledge systems in three different scenarios in an attempt to determine the conditions in which the Prometheus system may produce a higher quality knowledge base. The results of these experiments are presented along with some discussion and areas for future work.
Acknowledgements

I would like to thank my advisor, Mark Wineberg for his guidance in my work on this project and the opportunity to do so. Additionally, I would like to thank Deb Stacey for stepping up to assist me when Mark took leave to tend to his growing family.

I would also like to thank my friends and colleagues in the Web Solutions cluster of Campus Computing Services at the University of Guelph who supported me and encouraged me to take time from work to finish this project after far too much time away.

Of course, I must also give thanks to my friends who acted as a much needed lifeline to the outside world and also encouraged me to take time for myself on occasion.

I would have never reached this point in my life without the love and encouragement of my family. I am tremendously lucky to have two parents who believe in me and a brother who is my constant companion through all of life’s challenges.

Finally, I must thank my ever-supportive partner Elisabeth for her encouragement along with countless home cooked meals, baked goods, and the brainstorming sessions that were required to accomplish this project. Without your contributions this would not have been possible.
3.4.2.1 The User Cloud ........................................ 41
3.4.2.2 The Knowledge System Interface ......................... 45
3.4.2.3 Model of the Wiki System ................................ 46
3.4.2.4 Model of the Prometheus System ....................... 46
3.4.2.5 Parameters .............................................. 47

3.5 Experimental Design ............................................. 50

3.5.1 Experiment 1: Knowledge System Performance in a Typical Usage Environment ........................................ 50

3.5.1.1 Methodology ............................................. 50
3.5.1.2 Possible Outcomes ...................................... 53

3.5.2 Experiment 2: Knowledge System Performance in a Low Usage Environment ........................................ 53

3.5.2.1 Methodology ............................................. 53
3.5.2.2 Possible Outcomes ...................................... 54

3.5.3 Experiment 3: Knowledge System Performance in High a User Interaction Environment ........................................ 55

3.5.3.1 Methodology ............................................. 55
3.5.3.2 Possible Outcomes ...................................... 57

3.6 Data Analysis ...................................................... 57

4 Results ............................................................... 59

4.1 Experiment 1: Knowledge System Performance in a Typical Usage Environment ........................................ 59

4.2 Experiment 2: Knowledge System Performance in a Low Usage Environment 62
4.3 Experiment 3: Knowledge System Performance in High a User Interaction Environment ........................................ 65

4.4 Discussion of Results .................................................. 70
   4.4.1 Prometheus Design and Threshold Parameter Estimation ........ 74
   4.4.2 Simulation Framework Design and Parameters .................. 76
      4.4.2.1 Knowledge System Models ................................. 77
      4.4.2.2 The Simulation Framework ............................... 78
   4.4.3 Future Work ...................................................... 80

5 Conclusions ............................................................... 82

A Glossary ................................................................. 90
List of Tables

4.1 Initial statistics drawn from the values of the average knowledge base quality metric for the knowledge systems .................. 61
4.2 Initial statistics drawn from the values of the average knowledge base quality metric for the knowledge systems along with the difference obtained by subtracting the wiki value from the corresponding Prometheus value. ...... 62
4.3 Statistical summary of the data of the number of accepted changes for each knowledge system. ................................. 65
4.4 Statistical summary of the average knowledge base quality metric data for each knowledge system. ................................. 66
4.5 Knowledge quality statistics from experiments 1 and 3 for both the wiki and Prometheus knowledge systems. ...................... 67
4.6 Accepted change statistics from experiments 1 and 3 for the Prometheus knowledge system. ................................. 69
# List of Figures

2.1 The General Systems Framework ........................................ 15
2.2 The Four Component Framework ....................................... 16
2.3 The Knowledge Life Cycle Framework ............................... 17
2.4 Knowledge Management Practices Framework ....................... 17

3.1 C++ source code that defines the change class in the Simulation framework 29
3.2 The System Visit Event .................................................. 36
3.3 The access page event in the simulation framework .................. 37
3.4 The propose event in the simulation framework ....................... 37
3.5 The *judge page* event in the simulation framework ................ 38
3.6 C++ source code that defines the Knowledge Acquisition Simulation Framework class ........................................ 40
3.7 C++ source code that defines the User Cloud class .................. 42
3.8 Simulation parameter settings for experiment 1 ...................... 51
3.9 The PDF of the beta distribution with parameters \( \alpha = 10 \) and \( \beta = 15 \). ............................. 52
3.10 Simulation parameter settings for experiment 2 .................... 54
3.11 Simulation parameter settings for experiment 3 .................... 56
4.1 Wiki System: Average knowledge base quality over 50 runs. 

4.2 Prometheus: Average knowledge base quality over 50 runs. 

4.3 Wiki System: Histogram of average knowledge base quality over 50 runs. 

4.4 Prometheus: Histogram of average knowledge base quality over 50 runs. 

4.5 Wiki System: Average knowledge base quality over 50 runs. 

4.6 Prometheus: Average knowledge base quality over 50 runs. 

4.7 Wiki System: Histogram of average knowledge base quality over 50 runs. 

4.8 Prometheus: Histogram of average knowledge base quality over 50 runs. 

4.9 Wiki System: Number of changes accepted into the knowledge base over 50 runs. 

4.10 Prometheus: Number of changes accepted into the knowledge base over 50 runs. 

4.11 Wiki System: Average knowledge base quality over 50 runs. 

4.12 Prometheus: Average knowledge base quality over 50 runs. 

4.13 Wiki System: Histogram of average knowledge base quality over 50 runs. 

4.14 Prometheus: Histogram of average knowledge base quality over 50 runs. 

4.15 Prometheus System: Scatter plot of percent of accepted changes over 50 runs in experiment 1. 

4.16 Prometheus System: Scatter plot of percent of accepted changes over 50 runs in experiment 3. 

4.17 Prometheus System: Histogram of percent of accepted changes over 50 runs in experiment 1. 

4.18 Prometheus System: Histogram plot of percent of accepted changes over 50 runs in experiment 3.
Chapter 1

Introduction

Imagine a world in which every single person on the planet is given free access to the sum of all human knowledge. — Jimmy Wales, founder of Wikipedia

1.1 Wikis as Knowledge Management Systems

Knowledge management (KM) is a practice that has become common for both communities and organizations looking to support their members by providing an easily accessible authoritative source of common knowledge. The purpose of knowledge management practices is to provide a knowledge base so that others may benefit from the collective experiences of those who have come before them. A knowledge management system (KMS) can come in many forms but few have been as revolutionary as the computerized knowledge management system. Electronic knowledge management systems provide a cost effective means to allow high bandwidth usage of the knowledge base and have made the knowledge management system concept scalable by streamlining knowledge management practices and allowing concurrent usage.

Since their introduction in 1994, wikis have become a popular approach to the problem of collaborative knowledge management (KM). In recent years wiki use has increased greatly
and wikis have proven to be an effective tool in rapid knowledge base development. Wikis are strong solutions to the knowledge management problem because their open design allows anyone to contribute to the knowledge base with very little effort. Since the wiki design focuses on maintaining a low barrier of entry for potential knowledge experts the knowledge bases built using wikis can grow quickly without the need for oversight and intervention from the knowledge base administrators.

However, some researchers have expressed concerns that some of the elements of the wiki design can be considered a double edged sword. Critics of wikis point out that wikis are error-prone, lack a system of provenance, and are highly vulnerable to acts of vandalism by a vocal minority of users (16).

1.2 Prometheus: Controlling for Quality

In order to address some of these issues, this project suggests a modification to the knowledge acquisition component of the wiki system, dubbed Prometheus. By controlling the method in which knowledge is integrated into the knowledge base of a wiki-like system the goal of the Prometheus system is to demonstrate that the resultant quality of the knowledge base can be improved.

Prometheus aims to function as a wiki at heart by maintaining the low barrier to entry that makes the wiki such an effective tool for collaborative KM. Deviations from the standard wiki design are intended to address common critiques of the wiki system. Primarily, a threshold based user voting system is used to provide resilience against errors, vandalism, and low quality knowledge.

1.3 Overview

Critics of wiki systems in the role of knowledge management systems often cite the lack of knowledge base quality as a key characteristic that undermines the effectiveness of the KM
system. To address this problem this project presents the Prometheus system, a modification of the wiki design that attempts to control for knowledge quality before knowledge is added to the knowledge base.

In order to evaluate the effectiveness of the Prometheus design, this project will compare the performance of the Prometheus knowledge system against a traditional wiki knowledge system through simulation.

A simulation framework is presented which exposes models of the two knowledge systems to controlled operating environments and collects relevant data. In order to compare the knowledge systems three distinct operational environments will be used to explore the performance of the systems in typical and low use scenarios.

Knowledge system performance will be evaluated through the use of a newly proposed three dimensional knowledge quality model and an associated knowledge quality metric. Knowledge quality is modeled as a set of three values: truth, learnability, and conservatism. These values can be averaged to create a scalar representation of the knowledge quality.

Once the models have been exposed to the operating environment the data and resultant knowledge bases will be analyzed in order to determine the effects of the alternative knowledge acquisition strategy. Finally, this paper will discuss the conclusions that can be drawn from the analysis, the contributions of the project and possible directions for future work.

1.3.1 Terms and Definitions, The Glossary

For the reader’s convenience a glossary of terms is presented at the end of this document. All words appearing in the glossary will be expressed in boldface for their first occurrence in this document. The glossary will appear as Appendix A.
Chapter 2

Literature Review

2.1 Introduction

This section provides an overview of the central concepts that are necessary to the project and their state within the knowledge management and information technology literature today. First, the concept of knowledge is discussed with emphasis on the process of creating a suitable model for use with a knowledge quality metric. An introduction to knowledge management will follow with a focus on knowledge management systems (KMS). Finally, this section discusses the use of wikis as tools for knowledge management.

2.2 Knowledge

The challenge of defining the somewhat abstract concept of knowledge has received a considerable volume of attention from the KM community. Despite this, a great deal of variation still exists (13). The lack of a concrete definition of knowledge makes its use in research difficult. Fortunately, as (1) points out, a philosophical understanding of the concept of knowledge is valuable however it is not necessary for its use in KM or KBS research.

In KM research many definitions of knowledge take an organizational perspective such
as those found in (18; 31). These researchers define knowledge as a justified belief that enables an organization to take action. This definition incorporates several factors that highlight important characteristics of knowledge such as the concepts of justification and belief. Justification highlights the importance of relationships between individual pieces of knowledge. In addition to this the use of the word belief is important in that it highlights the relationship between knowledge, the user, and the truth of knowledge. Since a KM system lacks semantic understanding and advanced reasoning capabilities, the system must rely on the user’s belief in a piece of knowledge to determine truth. Thus, if the community believes that a piece of knowledge is true, then the system believes that it is true, regardless of the actual relation that exists between the knowledge and the real world. Through this it becomes evident that there is an intrinsic link between the users of the knowledge system and the system itself. Finally, the researchers state that knowledge acts as an enabler of action. In an uncertain world an entity cannot take action without knowledge and expect some rational result. Knowledge gives this entity the capability to act and to have some reasonable expectations about the consequences of its actions.

It seems advantageous to also consider knowledge from the more concrete, system based perspective that is used in the Information Technology (IT) literature. Within IT research it is common to define knowledge by distinguishing it from the concepts of data and information (1; 19; 41; 33). While there is slight variation in the community, most have settled on something resembling the following:

- **Data:** Raw numbers and facts.
- **Information:** Processed data.
- **Knowledge:** Authenticated information.

This definition by classification provides valuable insight. The concept of authenticated information suggests that the information has a historical provenance and that it has undergone a process of verification or validation. (41) provides a more concrete analogue of this
definition that serves as a solid foundation to work from. "(Knowledge) ... is information combined with experience, context, interpretation and reflection". This definition raises questions about the process of producing knowledge:

- What criteria defines the transition between these categories?
- What role does the user’s perspective play in the process of transition?
- Are all ‘authorized’ pieces of information created equal?

These questions suggest that an alternative approach to the process of modelling a piece of knowledge may be valuable. Rather than treating the abstract concept of knowledge in a binary fashion a fuzzy or probabilistic approach to seems beneficial. This approach promotes finer gradation in the knowledge model, allowing for a more sophisticated simulation.

2.2.1 Knowledge Types

At the highest level there are two basic types of human knowledge: tacit knowledge and explicit knowledge (30). Explicit knowledge is discrete, self contained, and can be effectively communicated directly through some formal language. An example of explicit knowledge might be a cookie recipe. In contrast to this, tacit knowledge is knowledge that is learned or understood and involves an element of human cognition. An example of tacit knowledge might be the ability to understand the consequences of a change to the recipe.

Unlike explicit knowledge tacit knowledge is extremely difficult to communicate directly. Typically, tacit knowledge must be communicated through a continual process of conversion to explicit knowledge, consumption, and cognitive activities to convert the knowledge back to tacit knowledge (3). Thus, knowledge systems deal in communicating both forms of knowledge, however the systems themselves deals exclusively with explicit knowledge.

Many researchers find value in categorizing knowledge based on the role that the knowledge plays (47). Common functional categories of knowledge include (13):
1. Declarative / descriptive: Knowledge describing what something is.

2. Procedural: Knowledge describing how to accomplish a task.

3. Causal: Knowledge that describes why something happens.

Some systems may behave differently based on which of these categories the knowledge fits into, applying different techniques for processing, storage, presentation, or metric generation. For the purposes of this project knowledge is assumed to be generalized explicit knowledge unless stated otherwise.

2.2.2 Knowledge Quality

In order to differentiate between KA methodologies it’s necessary to have a metric that can be used to compare the quality of knowledge in the resultant knowledge bases. The question of how to assess information quality (IQ) is a problem that the IT community has been facing for some time. While the concept of information quality seems simple at first glance, the problem is much more complicated in practice.

IQ metrics typically fall into one of two categories, objective and subjective (37). First, objective metrics that measure quality through comparison against established requirements or to conformance with real world observations. Examples of objective metrics may include accuracy, accessibility, or timeliness. In contrast to this, subjective metrics consider the perspective from which the information is used, usually through user feedback. Examples of subjective metrics include usability, presentation, and relevancy.

Objective metrics are typically easier to automate, which makes them good candidates for computer systems. Subjective methods typically require user involvement and thus can suffer from inconsistencies and a lack of rigour. Despite this, many consider subjective methods to be an important indicator of information system success. As (37) points out, neither approach is without its drawbacks and so an IQ methodology that combines both may be best.
Traditionally, many researchers considered accuracy as the sole dimension in data quality (27). Unfortunately, this approach seems naive as it neglects the important relationships that exist between the data, system, user, and the world. Much of the initial research into data properties for use in quality metrics was conducted by (46). In this exploratory study the authors independently followed a process that is very similar to the framework proposed by (20). This project was organized into the three following phases:

Phase 1: Identification of possible data properties.

Phase 2: Ranking the properties found in phase 1 based on perceived importance.

Phase 3: Grouping the significant properties found in phases 1 and 2 into data quality dimensions.

Phase 1 consisted of a survey sent to professionals in the knowledge management field. At its conclusion, this phase produced a detailed list of 179 data properties which were then ranked by knowledge workers during phase 2. Finally, phase 3 grouped these properties into 4 data quality dimensions:

Intrinsic: Incorporates properties such as believability, accuracy, objectivity, and reputation.

Contextual: Incorporates properties such as relevancy, timeliness, completeness, and appropriate amount of data.

Representational: Incorporates properties such as interpretability, ease of understanding, consistency, and concise representation.

Accessibility: Incorporates properties such as accessibility and access security.

While the full list of 179 properties is not complete it represents a diverse collection of possible properties. Further, the aggregated quality dimensions are well suited for work using simulation. Grouping the properties together benefits use in simulation by reducing the complexity of the knowledge model.
2.2.3 Knowledge Quality Frameworks

Initially many researchers (46; 48; 11) proposed unified frameworks that could be applied across all implementations equally. Many of these frameworks resemble the following framework proposed in (25), with information quality (IQ) dimensions grouped into categories:

<table>
<thead>
<tr>
<th>Reference</th>
<th>Process</th>
</tr>
</thead>
<tbody>
<tr>
<td>Functionality</td>
<td>Suitability, Accuracy, Interoperability, Compliance, Security, Traceability</td>
</tr>
<tr>
<td>Reliability</td>
<td>Maturity, Fault tolerance, Recoverability, Availability, Degradability</td>
</tr>
<tr>
<td>Usability</td>
<td>Understandability, Learnability, Operability, Luxury, Clarity, Helpfulness, Explicitness, User-friendliness, Customisability</td>
</tr>
<tr>
<td>Efficiency</td>
<td>Time behaviour, Resource behaviour</td>
</tr>
<tr>
<td>Maintainability</td>
<td>Analysability, Changeability, Stability, Testability, Manageability, Reusability</td>
</tr>
<tr>
<td>Portability</td>
<td>Adaptability, Installability, Replaceability, Conformance</td>
</tr>
</tbody>
</table>

However, some researchers have expressed the idea that a global set of IQ metrics is unlikely to meet the needs of every knowledge system (27; 17). Thus, several frameworks have been proposed to aid in the development of a quality model that is effective for a given system. (26) proposes the following:

1. Identify the user

2. Identify the metric application(s) (the applications and process that make up the system)

3. Identify the dimensions to be assessed

4. Prioritize the dimensions to be assessed by applying an Importance, Urgency and Cost metric to each dimension.
5. Develop specific assessment metrics for prioritized dimensions

The strength of this framework lies in its consideration of the context in which the system and its users are operating. As we know, in any knowledge system involving users the quality of the knowledge can vary widely based on the context in which the user is operating (36).

Similarly, (37) proposes the following framework:

1. Defining quality categories, covering both the objective product and subjective service quality views,

2. Determining the derivation method to use for criteria in each category based directly on the definition of that category, which effectively provides an automatic and natural classification of criteria into categories,

3. Deriving the criteria for the objective product quality component(s) of the framework,

4. Deriving the criteria for the subjective service quality component(s) of the framework, and

5. Empirically refining the criteria, especially subjective criteria, using focus groups. Note that this step does not involve any re-classification of criteria, since a sound basis for criteria classification is established based on category definitions as described in step 2.

These frameworks provide a solid position from which a series of metrics can be developed for use in the evaluation of knowledge bases resulting from differing knowledge acquisition methodologies.
2.3 Knowledge Management and Knowledge Management Systems

The term knowledge management was first coined in (32). In their book, Nonaka and Takeuchi posited that the competitive advantage that Japanese companies possessed over their Western counterparts was due to their knowledge management capabilities. Building on their work many researchers have proposed potential definitions of knowledge management.

Common to many of the definitions is the idea that knowledge management is a continual process. Much of the variation between the definitions arises from the division of the stages that compose the knowledge management process. This variation is illustrated in the following table, adapted from (19):
<table>
<thead>
<tr>
<th>Reference</th>
<th>Process</th>
</tr>
</thead>
<tbody>
<tr>
<td>(22)</td>
<td>Initiation, Generate, Repository, Distribution, and Transfer, Use, Retrospect</td>
</tr>
<tr>
<td>(1)</td>
<td>Creation, Storage and Retrieval, Transfer, Applications</td>
</tr>
<tr>
<td>(34)</td>
<td>Acquisition, Organize, Disseminate, Application</td>
</tr>
<tr>
<td>(4)</td>
<td>Discovery, Acquire, Creation, Storage and organization, Sharing, Use and Apply</td>
</tr>
<tr>
<td>(42)</td>
<td>Identify, Acquire, Preparation, Allocation, Disseminate, Usage, Retention</td>
</tr>
<tr>
<td>(23)</td>
<td>Acquisition, Coordination and Induction, Transmission and Diffusion, Creation</td>
</tr>
<tr>
<td>(43)</td>
<td>Organization and Retention, Creation and Acquisition, Dissemination, Utilization</td>
</tr>
<tr>
<td>(35)</td>
<td>Creation and Generation, Storage and Retrieval, Transfer, Application, Roles and Skills</td>
</tr>
<tr>
<td>(14)</td>
<td>Process about Knowledge, Process for Knowledge, Process from Knowledge</td>
</tr>
<tr>
<td>(9)</td>
<td>Identify Knowledge, Capture, Select, Stored, Service</td>
</tr>
<tr>
<td>(29)</td>
<td>Relate value, Acquire, Organize, Enable, Reuse, Transfer</td>
</tr>
</tbody>
</table>

While many of the above processes initially seem quite different, there are several commonalities between the component processes. The following chart groups the process elements from the above sources into common groupings based on their semantic meaning. The first group encompasses processes wherein knowledge is acquired and integrated into the system whereas the second group codifies the knowledge into a format that is easily manipulated by the system. Group three contains processes related to storage and group four encompasses processes related to knowledge dissemination. In many knowledge systems, a fifth process group is present wherein the existing knowledge base undergoes maintenance as the system operates.
<table>
<thead>
<tr>
<th>Semantic Group</th>
<th>Process Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Knowledge Acquisition</td>
<td>Initiation, Discovery, Identify, Generation, Creation, Acquisition, Process About Knowledge, Capture, Select</td>
</tr>
<tr>
<td>Knowledge Codification</td>
<td>Organize, Preparation, Enable Reuse, Modeling, Co-ordination</td>
</tr>
<tr>
<td>Knowledge Storage</td>
<td>Storage, Repository, Allocation, Induction, Retention</td>
</tr>
<tr>
<td>Knowledge Dissemination and Maintenance</td>
<td>Retrospect, Service, Distribution, Transfer, Dissemination, Sharing, Transmission, Process from Knowledge, Diffusion, Application, Use</td>
</tr>
</tbody>
</table>

By using the information presented in the above tables a practical definition of a knowledge management system can be created: A knowledge management system is any system that facilitates the processes of acquisition, codification, storage, dissemination, and maintenance of knowledge in a knowledge base. This is the working definition of a KMS that will be used through the remainder of this paper.

### 2.3.1 Knowledge Acquisition

The knowledge acquisition component is the gatekeeper of any knowledge management system. It acts as the primary channel through which new knowledge is added to the knowledge base.

Initially, many knowledge management systems acquired their knowledge by extracting the knowledge from a knowledge expert. This process often involved a knowledge engineer who facilitates this extraction process to ensure that the knowledge is properly formatted and is of high quality. Typical approaches to this process include the use of interviews, surveys, and guided knowledge transfer sessions.

Other researchers use artificial intelligence or data mining techniques to generate knowl-
edge from data sources. This approach has seen success in domains such as the credit industry where the area of study is small, well understood, and large amounts of data are available (44). However, as with all techniques that rely on computer systems these techniques are not perfect and can be unreliable.

(44) identifies several problems that commonly affect knowledge acquisition techniques:

Bandwidth Many approaches to knowledge acquisition are low bandwidth approaches. These approaches have some limiting factor that imposes restrictions on the rate at which knowledge can be added to the knowledge management system.

Latency Some knowledge acquisition strategies necessitate delays between the time of the knowledge acquisition the time that the knowledge is available for use by end users.

Inaccuracy Acquired knowledge can contain mistakes. Often, these mistakes are difficult to address and thus create challenges for knowledge maintenance. Further, the knowledge maintenance process itself can introduce errors that were not initially present.

### 2.3.2 KMS Research Frameworks

Several research frameworks have been proposed to aid researchers in understanding knowledge management systems. These frameworks help researchers to define the research domain, identify the important factors in KM systems, and to highlight gaps in the existing research. Since each framework provides a new perspective on KM research, a familiarity with each is beneficial.

The general systems framework views the knowledge system as a typical computational system which takes input, performs some processes, and produces some type of output. This perspective is beneficial because its generalized nature allows for the application of general system based concepts, techniques and metrics (13). The general systems framework is also very inclusive by nature, thus allowing it to be applied to research of a wide range of KMS
systems despite variance in KMS design and function. However, the generalized approach has drawbacks because many of the KM specific details are abstracted away. For example, use of the general systems framework could make it difficult to study the role that the various components (interface, analysis, storage, processes, etc) play in a KMS.

Based on the work in (10) the four component model of KMS departs from the system based perspective of the general systems framework to focus on the interactions between the user and the knowledge system. In this model, the knowledge system is composed of four primary components: the knowledge base subsystem, the user interface subsystem, the communication support subsystem, and the user. This framework places emphasis on the role of the user in the system as a generator and consumer of knowledge. Any research into the role of visualization, interface design, user interactions or knowledge transfer would benefit greatly from the use of the four component framework.

The knowledge lifecycle framework is more knowledge centric than the other frameworks. Developed by (39), it divides the knowledge lifecycle into four distinct stages that include creation, codification and storage, dissemination, and use. This framework allows KM strategies for each stage to be studied. This is also the only framework that explicitly captures the cyclical nature of the knowledge management system, allowing the knowledge base to be studied at various points in its lifespan.

The final KM research framework is the knowledge practices framework. This framework focuses on knowledge management systems that are designed to support organizational knowledge processes (13). The framework takes the form of a two dimensional matrix.
that categorizes knowledge management problems into one of four processes along two dimensions. The first dimension is the problem class, wherein the problem being solved is classified as either new or previously solved. The second dimension is the problem process which categorizes the problem as being related to problem recognition or problem solving. Using this matrix, KM problems are categorized into one of the following four categories:

1. Encouraging Serendipity - Knowledge practices that encourage the identification of new problems through creativity.

2. Mentoring and Training - Practices that support knowledge transfer.

3. Knowledge Creation - Practices that solve new problems through knowledge creation and storage.

4. Knowledge Acquisition - Practices that apply existing knowledge to address previously
2.3.3 Evaluation of KM Systems

With a wide variety of KM systems in use by organizations across the world it was inevitable that researchers would quickly find a need to evaluate the effectiveness of KM systems. Many researchers believe that KMS success can be measured through its effects on the operation of an organization (28; 5). Metrics used in these frameworks often include cost, benefits, employee performance, productivity or system use. While these techniques may be useful for some applications it requires that the system be fully implemented and integrated solved problems.
into a business environment. Further, this requires that the KMS researcher is required to be knowledgeable about the complex relationships between the knowledge system and the performance of the business, employees and other market factors. Additionally, this approach may not be consistent across organizations since each organization may have a different idea of what defines success for their specific operational environment.

Other researchers place a significant amount of emphasis on the interaction between the user and the knowledge system. This concept is often referred to as "knowledge use" and its use as a metric is often supported by the argument that "KM systems are ineffective if they are not used" (21). This argument is irrefutable, however it again requires that a working system be in operation and as such does not apply to research using simulated systems with no actual user involvement.

Since many evaluation frameworks for KM systems incorporate metrics that are not applicable to the evaluation of KA alone they are not applicable to the evaluation of a KA strategy. Thus, in order to accurately evaluate our KA methodologies a more generic methodology must be used. An evaluation mechanism that focuses on the quality of the accumulated knowledge is a promising alternative.

### 2.4 Wikis

A wiki is a set of interconnected web pages that allow for collaborative management of a knowledge base. Initially created by Ward Cunningham in 1994, the wiki has become a viable option for use as a KMS (16).

#### 2.4.1 Principals of Wiki Design

The design of the wiki system was based on the following 11 principals as defined by Ward Cunningham (45):
Principal               Explanation
Open                    If a page is found to be incomplete or poorly organized, any reader can edit it as he/she sees fit.
Incremental            Pages can cite other pages, including pages that have not been written yet.
Organic                 The structure and text content of the site is open to editing and evolution. A small number of (irregular) text conventions will provide access to the most useful (but limited) page markup.
Universal              The mechanisms of editing and organizing are the same as those of writing so that any writer is automatically an editor and organizer.
Overt                   The formatted (and printed) output will suggest the input required to reproduce it. (E.g., location of the page.)
Unified                 Page names will be drawn from a flat space so that no additional context is required to interpret them.
Precise                 Pages will be titled with sufficient precision to avoid most name clashes, typically by forming noun phrases.
Tolerant               Interpretable (even if undesirable) behavior is preferred to error messages.
Observable             Activity within the site can be watched and reviewed by any other visitor to the site.
Convergent            Duplication can be discouraged or removed by finding and citing similar or related content.

2.4.2 Characteristics of Wiki systems

From these principals outlined above, Cunningham created the first wiki system. In their book *The Wiki Way: Quick Collaboration on the Web* Leuf and Cunningham describe the key characteristics of a wiki system as follows (24):

- A wiki invites all users to edit any page or to create new pages within the wiki Web site, using only a plain-vanilla Web browser without any extra add-ons.
- Wiki promotes meaningful topic associations between different pages by making page link creation almost intuitively easy and showing whether an intended target
• A wiki is not a carefully crafted site for casual visitors. Instead, it seeks to involve the visitor in an ongoing process of creation and collaboration that constantly changes the Web site landscape.

2.4.3 Merits of Wikis as KMS

Wikis have several characteristics that make them well suited to the role of knowledge management. Wikis have a very low barrier to entry which encourages system use and participation. Technical and non-technical users can easily add, edit, delete, and consume knowledge with very little effort. This low barrier to entry means that even novice users can participate in the development of the knowledge base.

Due to the low barrier to entry wikis are also a very rapid method of building a knowledge base. This is evidenced by the name ‘wiki’ which is derived from the Hawaiian term ‘wikiwiki’ meaning ‘quick’, ‘fast’ or ‘to hasten’ (16).

Wikis are conversational. This conversational nature means that wikis can better facilitate knowledge acquisition from groups of participants rather than a single well informed expert. This helps to reduce the bandwidth bottleneck that can occur when trying to acquire knowledge from a single expert user. Further, conversational knowledge acquisition systems are ad-hoc and thus incorporate organic prioritization of subject matter.

Wikis are iterative. As the knowledge base evolves, wiki systems track the changes that were made over time and allow users to arbitrarily revert changes. This creates a safe environment that is volatile over the short term but is robust over the long term.

2.4.4 Criticism of Wikis as KMS

While Wikis have had some adoption as a KM tool some organizations have opposed use of the wiki. (15) details one such case and outlines many of the issues that prevented the organization from adopting wiki technology for its KM practices.
One criticism highlighted in (15) is the decentralized nature of the wiki. Critics expressed concern that the decentralized nature would result in a lower quality knowledge base because of a lack of managerial oversight, direction, and a formal review process.

Another criticism was that the wiki system makes it too difficult to track authorship. Because of the iterative and communal nature of the wiki system concerns were expressed that it would be difficult to track users who were using the system in a positive manner. The critics believed knowledge workers would not use the system because there is no reward for their work. Further, critics expressed that this would be a concern when attributing specific pieces of knowledge to a single author when intellectual property rights were considered.

Finally, the critics stated that the wiki system was too vulnerable to vandalism. Wiki vandalism refers to users damaging the knowledge base by intentionally introducing errors or by deleting content.

2.4.5 Simulation of KM Systems

Evaluation of knowledge systems is often cited as a challenge due to the role that knowledge experts must play in the knowledge system(7). Work involving knowledge experts is often time consuming, expensive and difficult to control. A high degree of control is particularly important when performing comparison between knowledge systems. In this regard, use of real knowledge experts can be challenging because an expert cannot use more than one system in isolation without their experiences influencing future trials. Simulation offers a strong alternative to the use of knowledge experts. Through the use of simulated experts, researchers can exercise a high degree of control over the experimental conditions. Despite this the use of simulation in KMS evaluation is quite limited at this time.

Much of the use of simulation in KMS research is centered around decision support and rule based classification systems. The succinct, structured language of these systems make them particularly well suited toward use in simulation because it simplifies the task of modelling experts and their knowledge. The most prominent work in this area is done by a
very small handful of researchers whose work concentrates on the use of simulated experts in ripple down rule systems (6; 7; 8). While promising, the researchers have identified several limiting factors including a lack of validation with real world experts and a lack of generalizability (6).

Simulation of the knowledge systems themselves offers other benefits. This allows researchers to quickly iterate on design ideas and quickly test and compare system performance against other knowledge system models. This approach offers a high degree of flexibility and provides an opportunity for innovation in KMS design.

2.5 Summary

This thesis draws from several distinct disciplines, all of which seek to solve similar problems through their own approach. From the business world, the field of knowledge management investigates the role of knowledge systems in an organizational context. Distinct from this the more technical researchers in information systems field investigate the properties and behaviours of the knowledge systems themselves. Both these fields draw on the experience of philosophers and psychologists in their attempts to explore the nature of human knowledge.

Because of the system based perspective and the objective metrics of the information systems approach this thesis will tend to fall on the systems side of the spectrum. Despite this, influence from the other fields is a valuable aid in the development of a well rounded investigation.
Chapter 3

Methodology

This section describes the methodology that was used to investigate the relationship between knowledge acquisition strategy and the resultant knowledge base in wiki KM systems.

3.1 The Prometheus System

In many aspects, the Prometheus system functions identically to a traditional wiki system. Knowledge is still organized in an identical manner, with collections of knowledge grouped into interlinked articles, each article describing a particular subject matter. Functionally, consuming knowledge stored in the Prometheus system is identical to a traditional wiki system. Despite its many similarities, Prometheus does differ from traditional wikis in a few key aspects that can have a large effect on the development of the knowledge base. Prometheus uses a threshold based acceptance and rejection mechanism to filter out changes that are considered to be harmful to the overall quality of the knowledge base. This threshold mechanism is designed to increase the quality of the knowledge that is stored in the system by requiring other users to review a change before it is committed to the knowledge base.

This threshold mechanism comes into effect whenever a user wants to make a change to
the knowledge base. This includes any additions, deletions, or edits. Rather than committing the change instantly as would occur in a wiki system the user’s change is first proposed to the knowledge system. Following this, the Prometheus system leverages the power of its community to determine whether the change is accepted into the knowledge base or is rejected.

Users of the system must vote on a change before it can be accepted into the system. Each user has one vote to spend on each change. A vote for a change can have one of two values. First, a positive has a value of +1 and is a representation of the user’s support of that change. The second value is a vote in opposition to the change, which receives a value of -1. Once the user has votes on a change once they cannot vote again. All of the negative and positive votes for a change are summed to create a single integer that represents the state of the change, called the change value. Thus, if a change received three positive votes and two negative votes that change would have a value of $3(+1) + 2(-1) = +1$.

When a change is initially proposed to the knowledge system it is created with a change value of +1. This value represents the support of the author who created the change.

The Prometheus knowledge system uses two parametrized values that represent the upper and lower thresholds that must be exceeded in order for the system to take action on a change. If the value of a change exceeds the upper threshold the system will accept that change into the knowledge base and will begin to serve that change to all subsequent visitors to the page. If a change’s value falls below the lower threshold that change is rejected and the proposed change is deleted.

3.1.1 Page management

Any user in the system can create a page. Each page is initially created blank in the system and must be dedicated to one singular topic. Only one page is allowed to exist on each topic. Thus, the topic of the page functions as a globally unique identifier. Since pages by default have no content there is no voting process required for the creation of a page.
However, any additions to the content of the page must be submitted as proposed changes and these changes must exceed the value threshold to be accepted into the knowledge base.

Just like changes, users have the capability to vote in support of or in opposition to a page. Pages also have associated value that is compared against the parametrized threshold values. However, pages differ from changes in that they are always accepted into the knowledge base. Thus, as changes receive positive votes their value increases but the state of the page does not change. Conversely, if a page falls below the rejection threshold then that page is deleted from the system and a new page may replace that page at a later date.

### 3.2 Background

The design of the Prometheus system is intended to provide a higher quality knowledge base in a typical operating environment however the system may not perform better in all scenarios. The threshold based acceptance-rejection knowledge acquisition strategy means that Prometheus has a higher barrier to entry than a traditional wiki system that accepts all proposed changes to the knowledge base. This cost means that Prometheus may not be an appropriate choice for all KM environments. One particular area of concern for Prometheus may be environments which can be characterized as 'low use'.

A low usage scenario is any environment in which the volume of usage of the knowledge system may be too low for the knowledge system to function effectively. The design of the knowledge system has an impact on the volume of usage which can be characterized as a low usage environment. Thus, an environment that is considered low usage for Prometheus may not be too low for a traditional wiki system. Low usage environments in this project have been designed to starve the Prometheus wiki system while still providing a sufficient amount of usage for the traditional wiki system. Real world examples of low usage scenarios may include:

- Subject domains where there is a small number of experts that are capable of contributing.
• Environments where there is a small user base.

• Environments where experts lack the motivation to use the site on a regular basis.

• Environments in which access to the system is restricted by an external constraint such as time or geographical location.

Conversely, a typical usage scenario is an environment in which each system is exposed to a sufficient level of traffic to allow the system to function as intended. Thus, a typical usage scenario is an environment that cannot be classified as a low usage environment for either knowledge system.

In order to investigate the performance of the Prometheus system relative to the standard wiki system this project proposes several hypotheses that will be examined throughout the remainder of this project. These hypotheses focus on the use of the two systems in both typical and low usage environments:

1. Prometheus’ threshold system uses users to analyze and select for knowledge that is of high quality. Thus, in a typical usage scenario the quality of the resultant knowledge base in Prometheus should be higher than that of a wiki system operating in the same environment.

2. Prometheus’ threshold system raises the barrier to entry and slows the knowledge acquisition process. Thus, in a low usage scenario the threshold system may prevent Prometheus from acquiring knowledge that would otherwise be deemed acceptable. In this environment the traditional wiki system may have an equal or higher average quality of knowledge than the Prometheus system that was exposed to the same environment.

3. The threshold based knowledge acquisition system of Prometheus relies heavily on user feedback in order to accept high quality knowledge and to reject low quality knowledge. Thus, in a typical usage environment the Prometheus system may perform
better if users are more likely to contribute to the system by submitting changes to the knowledge base and judging the changes that other users have submitted to the knowledge base.

3.3 Knowledge architecture

3.3.1 Knowledge and Changes

The smallest entity in the simulation framework is a single piece of knowledge. Conceptually, each piece of knowledge represents a collection of grammatical sentences, data, graphs or diagrams composed with the intent to codify a singular piece of knowledge. However, the simulation framework does not require that these details are explicitly modelled. Rather, the framework is designed to deal with collections of generalized knowledge as opposed to a collection of explicit instantiations of knowledge. Since the simulation is agnostic with respect to the form or content of the knowledge, metadata is used to describe the knowledge content. By dealing exclusively with this metadata the simulation framework is able to abstract away the concrete details of the knowledge.

In the context of the simulation framework the term change is often used as a synonym to refer to a single piece of knowledge. A change is a piece of knowledge that may or may not have been accepted into the knowledge base. The term change is used because each additional piece of knowledge represents a deviation from the initial, blank state of the page. Thus, when a user attempts to add to or edit the knowledge in the knowledge base the user is said to be proposing a change to the knowledge base.

3.3.2 Knowledge Representation

Each change in the knowledge system has three primary properties that are used to determine the overall quality of a given piece of knowledge. In the simulation framework, each property may take on a value inside of a range from $[0,1]$. These properties are:
**Truth** This property represents the relationship that exists between the codified knowledge and the observable real world. This property encompasses knowledge quality dimensions such as accuracy, objectivity, and provenance.

**Learnability** This property represents the the relationship between the codified knowledge and the user. This property encompasses quality dimensions such as interpretability, accessibility, relevancy, timeliness, and completeness.

**Conservatism** This property represents the relationship that exists between the single piece of knowledge and the rest of the knowledge in the knowledge base. This property encompasses quality dimensions such as agreeability, consistency, and conformity.

As suggested by (26; 27; 37; 17) these quality dimensions were selected specifically for use in this project. The selection process roughly followed the procedures outlined in section 2.2.3, the knowledge quality frameworks. A number of sources (46; 11; 17; 20; 36) were surveyed to create a list of possible knowledge properties that could be used as indicators of knowledge quality in general purpose knowledge systems. This list was then prioritized according to the user and system characteristics and were grouped according common elements. These elements were then generalized to provide the final list of three quality dimensions: truth, learnability, and conservatism. While not identical, the quality dimensions selected for this work are conceptually similar to those used in (46).

Figure 3.1 is an excerpt of the C++ code which defines the minimal set properties and methods that a change must have to be used in the framework:

In addition to the truth, learnability, and conservatism properties, each change has two other boolean values that are used to represent the status of the change in the knowledge system. The deleted flag is set if the change has been flagged as deleted in the knowledge base. Treating deletion in this manner allows data about delete changes to be collected posthumously and allows knowledge systems to restore deleted changes if necessary. Similarly, the accepted flag indicates whether the change has been accepted into the knowledge
Figure 3.1: C++ source code that defines the change class in the Simulation framework base. This is an important consideration when collecting information about the quality of the knowledge base, as only accepted changes are included in the quality assessment.

### 3.3.2.1 Knowledge Quality

When considering knowledge quality, each of the above quality dimensions is given an equal weighting. Thus, within the framework knowledge quality, $Q_k$, will be determined as follows:

$$Q_k(k) = \frac{k_{truth} + k_{learnability} + k_{conservatism}}{3}$$

Where $k$ is an individual piece of knowledge. Essentially, this quality metric is the average of the three individual dimensions. Since each of dimension has a domain of $[0,1]$ the resultant quality metric is also restricted to the domain of $[0,1]$. 
3.3.3 Pages

Within wiki systems, individual pieces of knowledge are organized according to topical groupings. These groupings are often referred to as pages since each ‘page’ in the web based system is devoted to a single topic of discussion. Thus, for the remainder of this document the terms ‘page’ and ‘topic’ will be used interchangeably within the context of the knowledge system. For example, adding a page to the knowledge system is to be considered the action of adding a new topic that can be used as a grouping mechanism for individual pieces of knowledge.

3.3.3.1 Page Quality

By building upon the above formula for knowledge quality a straight forward measurement of page quality, $Q_p$, can be developed. Since a page is simply a collection of changes we can define the page quality metric to be the average quality value for all of the changes that have been proposed and accepted for the page. Note again that this quality metric falls in the domain of $[0,1]$.

3.3.4 Links

Each page in the system can have an arbitrary number of links to other pages in the system. In a sense, these links can be thought of as a representation of the ontological relations that exist between pieces of knowledge. However, this analogy cannot be considered entirely valid as simple links between topics lack the descriptive capacity to distinguish between different kinds of relationships. To address this, researchers often add context through the addition of semantic data and annotations(40).

We are not concerned about the presence of explicit links between pages in this project as their addition would add a considerable amount of complexity without adding much in terms of useful data. Instead, an analogy to the link mechanism, page popularity, will be used. Page popularity plays a key role in determining which page a user will visit when
they use the system. Since linking is the primary means in which users travel between pages it stands to reason that a page with high popularity will also have a high number of links pointing to it. Thus, we believe that this page popularity analogy is appropriate for use in this project.

3.3.5 The Knowledge Base

Much like a page is a collection of pieces of knowledge, the knowledge base is a collection of pages. Pages in the knowledge base may be marked as deleted. Once marked as deleted pages are not considered accessible unless a page is restored through the actions of a user or through some mechanism of the knowledge system itself.

3.3.5.1 Knowledge Base Quality

By further extending the ideas behind our knowledge and page quality metrics we can determine that the overall quality of the knowledge base $Q_{kb}$ can be measured by averaging the quality of all of the pages that have not been deleted in the knowledge base. Again, the domain for this metric falls in the range $[0,1]$. This metric will be the primary metric used to determine which knowledge system exhibits superior performance in each operating environment.

However, when considering the knowledge base as a whole the quality of the pages alone may not be sufficient to judge the success of the knowledge system. This may be particularly true in low usage scenarios where the Prometheus threshold system may prevent a significant number of the knowledge from being accepted into the knowledge base. Thus, the number of changes accepted and the proportion of accepted changes may be used as secondary metrics for evaluating the performance of the two knowledge systems.
3.4 Overview of The Simulation Framework

The simulation framework is a pluggable framework that provides minimal interfaces of the various components and handles the integration between them. These interfaces can be implemented and extended to provide additional functionality and behaviours that allow experimentation with models of different knowledge systems. The simulation framework provides a unique opportunity for researchers to rapidly prototype and evaluate the performance of knowledge management systems. This allows researchers to iteratively test and improve their designs in a specific operating environment.

The framework also provides a great deal of control over the environment in which the knowledge systems operate. With this control the simulation framework allows researchers to expose multiple knowledge systems to environments that have identical characteristics. This consistency allows researchers to directly compare multiple knowledge systems against one another without fear of the unseen factors and biases that are nearly impossible to avoid in large scale real world trials.

3.4.1 The Visit event

The primary function of the framework is to expose the knowledge system to a pre-defined number of visit events, co-ordinate the various activities that occur in each visit event, and log the appropriate data for analysis. The UML activity diagram of the visit event is provided in figure 3.2.

The first action that occurs in each visit is the generation of a new user. This functionality is handled by the User Cloud, which represents the global collection of users and their cumulative knowledge. Further detail on the role and functionality of the user cloud is provided in section 3.4.2.1. When a user is generated by the user cloud, the user is assigned a page to visit and a set of properties that represent the user’s capabilities. Much like a change, these properties are truth, learnability, and conservatism. These properties will be used as mean values for changes generated by the user and for determining the outcome of
interactions between the user and the knowledge system.

The user action decision is an important part of the visit event. This action is decided through the use of a joint probability distribution (JPD) that is provided as a runtime parameter. The system takes this JPD and converts it into a cumulative distribution function (CDF) for use in the decision making process. A uniform random number generator is used to generate a random number, which is then compared against the inverse of the CDF to determine the course of action that the user will take. The three choices of action are:

1. Access Page
2. Propose Change
3. Judge Page

The access page branch is purely a read operation from the perspective of the knowledge system. Within this branch an access page event is generated by calling the accessPage() method in the knowledge system. The access page event is outlined in figure 3.3.

The access event updates the user’s observed values of truth, learnability and conservatism with the values of the properties from the page that the user is accessing. The user’s observed values are assigned the average values of each property on the page.

In addition to updating the user’s observed values the access operation generates a user satisfaction value based on the differences between the observed values and the user’s core values. This value represents the user’s overall experience with the page and is defined as the probability that a user will have a positive experience given the observed page statistics. The user satisfaction probability is calculated as follows:

\[
\text{satisfaction} = \frac{\text{probToHelp}(ut, ot) + \text{probToHelp}(ul, ol) + \text{probToHelp}(uc, oc)}{3}
\]
Where

\( u_t \) is the user truth
\( o_t \) is the page truth
\( u_l \) is the user learnability
\( o_l \) is the page learnability
\( u_c \) is the user conservatism
\( o_c \) is the page conservatism

and \( \text{probToHelp}(u, o) \) is defined as:

\[
\text{probToHelp}(u, o) = 0.5 \times (o - u) + 0.5
\]

Where

\( u \) is the user’s value
and \( o \) is the observed value.

This user satisfaction value plays an important role in soliciting feedback from the user throughout the visit event. Satisfaction is particularly important in the user’s consideration of page deletion and addition.

Both the \textit{propose change} and \textit{judge page} branches start with an access event. This access event must occur because it is assumed that a user must first view the page in order to propose a change or to judge the content of the page. Without this, the user would lack the motivation or capability to act on the page without prior knowledge about its current state.

The add change event, presented in figure 3.4, generates a new change using the user’s core properties as well as the observed properties from the current page. The properties for the new change are generated using a set of pseudo-random number generator that is passed to the framework as a parameter. These values are then shifted so that the average value of the generated properties is the average of the user’s properties and the observed
properties. This newly generated change is then submitted to the knowledge system and the user expresses their support of their newly proposed change to finish the process.

The *judge change* branch is the most complex of the three branches. This branch simulates the act of the user reading through the content of the page and expressing their opinion on certain pieces of knowledge. A UML activity diagram of the judge event is presented in figure 3.5.

For each change, the user may choose one of the three following courses of action:

- Support the change.
- Express no opinion about the change.
- Oppose the change.

After the user’s action has finished execution, each user gets a chance to express their opinion about the page that they have been interacting with. If the user has a negative satisfaction the user has a probabilistic chance to express opposition for the page. Similarly, a user with a positive experience has a chance to express their support of the page. The probability to act on their experience increases as the user’s satisfaction deviates from a parametrized threshold value. The rate of this increase is defined through a linear equation in the format of $y = mx + b$ where $x$ is the independent variable, $m$ is a parameterized slope and $b$ is a parametrized $y$-intercept value.

Finally, the user is returned to the user cloud where stats are collected and the user generation properties for that page are updated with the data from the user’s experience. This process simulates knowledge dissemination and affects the properties of subsequent users who visit the same page.
Figure 3.2: The System Visit Event.
Figure 3.3: The access page event in the simulation framework.

Figure 3.4: The propose event in the simulation framework.
Figure 3.5: The *judge page* event in the simulation framework.
3.4.2 The Simulation Framework

The key components of the framework are described in further detail below. They are:

**The User Cloud** Responsible for user creation and deletion.

**The Knowledge System** A model of the knowledge system that is being visited by the users.

**The Writer** A logging component that collects data about the knowledge system and creates a series of output files that describe the state of the knowledge system throughout the simulation.

The core framework class facilitates the initialization of all of the components and parameters that are necessary to run the simulation. Figure 3.6 presents the C++ header file that defines the simulation framework class. The primary function of the simulation framework is to co-ordinate the interactions between the components of the system and the user as each visit event is carried out.

As stated above, the simulation uses a joint probability distribution (JPD) to choose the action that the simulated user will perform when they visit the system. The setJPD() methods allows the values of the JPD to be set. Each value represents the probability of the corresponding event occurring. Note that since these values are used as a JPD, each value must fall in the range of [0,1] and all of the values must sum to exactly 1. The three parameters are as follows:

- **p.a** The probability take the *access branch* of the user action decision.
- **p.p** The probability to take the *propose change* branch of the user action decision.
- **p.j** The probability to take the *judge page* branch of the user action decision.
Figure 3.6: C++ source code that defines the Knowledge Acquisition Simulation Framework class

The setBias method allows a bias to be applied to the changes generated by the system. This function sets the bias parameter which is used as a co-efficient when the truth, learnability, and conservatism properties of the change are generated. Through this mechanism, researchers can influence the general quality of the generated knowledge. Values for the bias factor are stored in a variable of type double, though negative values should not be used since negative bias numbers will always result in the generation of a zero value for that property.
The simulation framework also has several properties that are used as thresholds and modifiers for the page support and opposition probability generation. For both support and oppose probability generation three parameters are used:

1. A threshold value.
2. A slope value.
3. An intercept value.

Much like the user satisfaction value, these parameters define a line that is used to find a probability based where the independent variable is the difference between the threshold value and the user’s satisfaction value.

Finally, the run method runs the simulation for a given number of repetitions, which is passed as a parameter of type integer. This value must be positive. Each repetition represents one visit event wherein a single user visits the knowledge system.

3.4.2.1 The User Cloud

The user cloud is responsible for generating users as well as for the maintenance of a global user knowledge base. Figure 3.7 presents the C++ code that defines the user cloud class.

The genUser method generates a new user for use in the simulation. This method takes one argument, which sets the bias factor that his user will use when generating a new change. When a new user is generated a user object is created and the user is probabilistically assigned a page id from the user knowledge base.

Page selection is influenced by a page popularity ranking that is designed to mimic the power law link distribution that has been observed on the Internet (12). Each time a page is added to the user knowledge base, a link is added to another page using the preferential attachment methodology wherein the pages with high popularity are more likely to receive
Figure 3.7: C++ source code that defines the User Cloud class

```cpp
class User_Cloud {
public:
  User_Cloud(User_KB* u, Generator* g, Gen_Set* pGen, Gen_Set* uGen);
  ~User_Cloud();
  void retUser(User* u);
  User* genUser(double bias);
  double probToAddU(double s);
  double probToAddL(double s);
  double probToDel(double s);
  vector<string> print();
  User_KB* ukb;
  Gen_Set* p_gen;
  Gen_Set* u_gen;
  Generator* gen;

  // Parameters for page addition and deletion.
  double t_add_u;
  double add_u_m;
  double add_u_b;
  double t_add_l;
  double add_l_m;
  double add_l_b;
  double t_del;
  double del_m;
  double del_b;

  // stats
  int generated;
  double avg_truth;
  double avg_learn;
  double avg_cons;
  double avg_sat;
};
```

the link than pages with low popularity. Each time a link is added to a page, the popularity of that page increases by one. This approach creates a power law link distribution (2).

Once a page has been selected for the user values for truth, learnability, and conservatism are generated. These values are generated using a pseudo random number generator and shifted so that the mean value is equivalent to the mean value for the corresponding user page. The set of generators that are used for generating truth, learnability, and conservatism are assigned at run time and can draw from any distribution. For all work within this project a beta distribution with $\alpha = 12$ and $\beta = 12$ is used for user property generation.

In addition to user generation, the user cloud is also responsible for the maintenance
of a global user knowledge base. This knowledge base is necessary because any knowledge system does not exist in isolation and is not the sole channel of communication between users. Knowledge transfer occurs through other means including direct communication between users, or through the use of separate knowledge systems. Thus, a user knowledge base is used to update the user generation values based on the experiences of the users that have previously existed.

After each visit event, the user is returned to the cloud via a call to the retUser() method and the global values for the page in the user knowledge base are updated. Each value is updated by adding half the difference between the observed and the current value, normalized for the population of users:

$$new\text{\_value} = cv + \frac{(ov - cv)/2}{nu}$$

Where

- $ov$ is the user’s observed value
- $cv$ is the current page value
- and $nu$ is the number of users in the user cloud

These updates influence the properties of the users generated for the page in future visit events.

Further, the user knowledge base serves to regulate the popularity of topics in the global user consciousness. In effect page management in the user knowledge base becomes an analogue for trends in the global user consciousness. Page management in the context of the user knowledge base determines when topics are added or removed from the global user consciousness. User knowledge base page addition and deletion does not directly have an impact on the knowledge system knowledge base, but rather effects the chances of users visiting pages on the topic represented by that page.
After each user visit, the user has a probabilistic chance to add or remove a page in the knowledge base. Again, this chance is determined using the user satisfaction in conjunction with parametrized threshold, slope, and intercept values. Page addition has both an upper and a lower threshold values while page deletion only uses a lower threshold. The two page addition thresholds are used to take two motivations for page addition into consideration. First, the user may be very satisfied with the page and thus is inspired to add another page on a related topic. Alternatively, the user may be very unsatisfied with the page and may want to create a separate page intended to supersede the original page. At this point the user can perform one of three actions:

1. Add a new page.

2. Delete the page that was assigned to the user

3. Do nothing.

Once a page has been created in the user cloud knowledge base, it’s possible for that page to be selected as the page for a subsequent user visit. The page will be added to the knowledge system on its first visit. When this visit occurs the new page properties will be generated using the knowledge properties of the user who originally triggered the addition of the page in the user knowledge base.

Similarly, page deletion removes the page from the pool of pages that can be assigned to new user. In essence this page has fallen out of favour of the global community of users. Thus, the page will still exist in the knowledge system but it will become static.
3.4.2.2 The Knowledge System Interface

The simulation framework developed for this project provides a minimal interface that models of knowledge systems must implement. These operations serve as event triggers from the perspective of the knowledge system. The framework defines the following operations:

**Access Page** The user accesses a single page in the KS.

**Propose Change** A user attempts to change the knowledge on a page in the KS.

**Judge** A user judges the knowledge on a single page in the KS.

**Support Change** A user supports a single piece of knowledge in the KS.

**Oppose Change** A user condemns a piece of knowledge in the KS.

**Support Page** A user supports a page in the KS.

**Oppose Page** A user condemns a page in the KS.

While the interface that the knowledge systems must implement is defined by the framework, the implementation of that interface is not. By varying the way in which each of the methods is implemented researchers can create models of a wide variety of knowledge systems. For use in this project, two models were developed. The first was a traditional wiki system and the second was a modification on the wiki design called Prometheus. The primary way in which Prometheus deviates from the standard design is through a threshold and voting based system that is used to validate proposed changes to the knowledge base.

The following sections describe the models of the wiki and Prometheus knowledge systems that will be used in conjunction with the simulation framework. Some aspects of the knowledge systems have not been modelled in favour of simplicity. In these cases, the omitted functionality is identical in both systems. Thus, the omission will not affect the relative performances of the two systems.
3.4.2.3 Model of the Wiki System

The defining characteristics of each wiki model is the manner in which it reacts to the support and oppose events. As discussed previously, wiki systems typically allow anyone to edit the knowledge base and offers no resistance to modification. In the wiki model the support change event causes the knowledge system to immediately accept a newly proposed change into its knowledge base. If the change has already been accepted, then this operation does nothing.

Similarly, the corresponding oppose change event immediately marks a change as deleted. All subsequent visits to the page will ignore the change and the properties of the change will not affect the knowledge quality metric.

At the page level the support page event has no effect while the oppose page event has a chance of resulting in the deletion of the page. Since deletion is an action with significant consequences, the oppose page event does not always result in page deletion. Instead, a parameter is used to determine the probability that a page will be deleted when a user opposes it. If a page is deleted this deletion occurs in the knowledge system and not in the user cloud. Thus, a subsequent user may return to the page and re-create it.

3.4.2.4 Model of the Prometheus System

The primary modification of the wiki design that the Prometheus system implements is the addition of a threshold system for change addition and deletion. When a change is proposed to the knowledge system a corresponding vote record is created for that change. As the change receives support and opposition, the vote record tracks the status of the page as an integer number where a support event represents a value of +1 and an oppose event represents a value of -1. Each time a user votes on a change, the vote record is compared against two parametrized threshold values. If the vote record passes the positive threshold, the change is accepted to the knowledge base and it’s properties will be factored into the calculation of the overall page properties. If a change exceeds the lower threshold, that
change is considered to be rejected and will no longer be available for access or judgement.

A similar approach is taken to page deletion. Pages in Prometheus also maintain a record of votes. As the page receives positive and negative support, this value is checked against a lower threshold for page deletion. Thus, the page is only deleted once the page has received enough opposition to surpass the deletion threshold.

These threshold values help Prometheus to be more robust against the whims of individual users who are a vocal minority. While the threshold values slow down the rate at which new knowledge can be added to the knowledge base they also protect from the changes that have a negative effect on the quality of the knowledge base.

Currently, the threshold values are passed to Prometheus as initial parameters. Parameter selection poses a challenge, as threshold values that may be appropriate in one environment may not be appropriate in another. In future work, dynamic threshold values that change with the activity or popularity of the page or system user base may prove to be a more effective strategy.

### 3.4.2.5 Parameters

The simulation framework provides a large number of parameters that can be used to change the environmental conditions that the knowledge systems operate in. This section presents a list of all of the parameters that can be tuned within the simulation framework.

**num_users** A positive integer number representing the number of users in the user cloud.

This parameter affects the influence that each user visit has on the user population as a whole.

**num_pages** A positive integer number representing the maximum number of pages that can exist in the system.

**num_visits** A positive integer value that determines the number of visits that occur in each run of the simulation.
**num_reps** A positive integer value that determines the number of replicates to use in the simulation.

**bias** A real number value that functions as the bias co-efficient to use when generating truth, learnability, and conservatism values for a change.

**access_probability** A real number in the domain [0,1]. This value is the probability that the *access page* branch will be selected for the user action decision in the visit event.

**propose_probability** A real number in the domain [0,1]. This value is the probability that the *propose change* branch will be selected for the user action decision in the visit event.

**judge_probability** A real number in the domain [0,1]. This value is the probability that the *judge page* branch will be selected for the user action decision in the visit event.

**t_add** A real number value in the domain [0,1]. The user satisfaction upper threshold for the generation of page addition probabilities.

**add_m** A real number value. The slope used in the generation of page addition probabilities for the upper threshold.

**add_b** A real number value. The intercept value used in the generation of a page addition probabilities for the upper threshold.

**t_add_l** A real number value in the domain (0,1). The user satisfaction lower threshold for the generation of page addition probabilities.

**add_l_m** A real number value. The slope used in the generation of page addition probabilities for the lower threshold.

**add_l_b** A real number value. The intercept value used in the generation of a page addition probabilities for the lower threshold.

**t_del** A real number value in the domain (0,1). The user satisfaction threshold for the generation of page deletion probabilities.
**del_m** A real number value. The slope used in the generation of page deletion probabilities.

**del_b** A real number value. The intercept value used in the generation of a page deletion probabilities.

**user_page_beta_alpha** The $\alpha$ shape parameter of the beta distribution used in the generation of initial page properties in the user cloud. This value must be a positive, real number.

**user_page_beta_beta** The $\beta$ shape parameter of the beta distribution used in the generation of initial page properties in the user cloud. This value must be a positive, real number.

Similarly, each model exposes some parameters of its own. The wiki model provides the following:

**deletion_probability** The probability that a page will be deleted on a page opposition event.

While Prometheus provides some tuning parameters for the threshold system:

**threshold_upper** An integer value that must be greater than the threshold_lower value. The upper support threshold for changes and pages.

**threshold_lower** An integer value that must be lower than the threshold_upper value. The lower opposition threshold for changes and pages.
3.5 Experimental Design

Using the simulation framework described above, this project tests the three proposed hypothesis outlined in section 3. In each experiment the two knowledge systems will be exposed to the same operational environment. After running the simulation data will be collected and analyzed to determine the result of the test.

3.5.1 Experiment 1: Knowledge System Performance in a Typical Usage Environment

This experiment is designed to test hypothesis 1:

"Prometheus’ threshold system uses users to analyze and select for knowledge that is of high quality. Thus, in an average usage scenario the quality of the resultant knowledge base in Prometheus should be higher than that of a wiki system operating in the same environment."

3.5.1.1 Methodology

Each system will be run in an identical operational environment over 50 runs of the simulation. The simulation parameter settings for this experiment are listed in figure 3.8.

With 1000 users taking part in the simulation and 10000 visit events in each run, each user will visit the system an average of 10 times throughout the duration of the evaluation period. This should allow the users ample opportunity to utilize system and should provide adequate usage to overcome the barrier to entry that results from Prometheus’ threshold based acquisition system.

Other notable parameter settings include the probabilities for page access, change and judge branches of the visit event. Best guess efforts for these parameter settings resulted in 40% chance to access the page alone, 25% chance to propose a change to the page, and
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Wiki System</th>
<th>Custom System</th>
</tr>
</thead>
<tbody>
<tr>
<td>num_users</td>
<td>1000</td>
<td>1000</td>
</tr>
<tr>
<td>num_pages</td>
<td>10000</td>
<td>10000</td>
</tr>
<tr>
<td>num_visits</td>
<td>10000</td>
<td>10000</td>
</tr>
<tr>
<td>num_reps</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>bias</td>
<td>1.025</td>
<td>1.025</td>
</tr>
<tr>
<td>access_prob</td>
<td>0.4</td>
<td>0.4</td>
</tr>
<tr>
<td>propose_prob</td>
<td>0.25</td>
<td>0.25</td>
</tr>
<tr>
<td>judge_prob</td>
<td>0.35</td>
<td>0.35</td>
</tr>
<tr>
<td>t_add</td>
<td>0.6</td>
<td>0.6</td>
</tr>
<tr>
<td>add_m</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>add_b</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>t_add_l</td>
<td>0.3</td>
<td>0.3</td>
</tr>
<tr>
<td>add_l_m</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>add_l_b</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>t_del</td>
<td>0.4</td>
<td>0.4</td>
</tr>
<tr>
<td>del_m</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>del_b</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>user_page_beta_alpha</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>user_page_beta_beta</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>(wiki) del_prob</td>
<td>0.2</td>
<td>–</td>
</tr>
<tr>
<td>(Promethens) threshold_upper</td>
<td>–</td>
<td>5</td>
</tr>
<tr>
<td>(Promethens) threshold_lower</td>
<td>–</td>
<td>5</td>
</tr>
</tbody>
</table>

Figure 3.8: Simulation parameter settings for experiment 1.

35% chance to judge the content of the page. These parameters were chosen under the assumption that:

\[ \text{probability to read} > \text{probability to judge} > \text{probability to propose a change} \]

In addition to this, note that these parameter settings are assumed to be constant throughout the simulation, whereas these probabilities would likely change over time in a real world scenario.

A beta distribution with parameters \( \alpha = 10 \) and \( \beta = 15 \) were chosen for the initial values of the truth, learnability, and conservatism properties of pages. Figure 3.9 shows the probability density function of this function. The beta distribution was chosen because its flexibility allows for a high degree of variability which simplifies implementation. The specific parameter settings were chosen in order to create an asymmetric distribution which favours probabilities in the mid to low range. With an expected value of 0.4, these low...
initial values were chosen to allow room for the iterative processes of the knowledge systems to work.

![Beta Distribution](image)

Figure 3.9: The PDF of the beta distribution with parameters $\alpha = 10$ and $\beta = 15$.

The threshold values of 5 and -5 for the Prometheus system were chosen as reasonable values that are intended to provide some resilience against low quality knowledge while still maintaining a low barrier to entry. These values were chosen after an exploratory study was used to determine effective parameter settings by running the system through a constant environment in which the threshold values were varied from 2 to 15 for the upper threshold and -2 to -15 for the negative threshold. Asymmetrical threshold values were not considered though they may be an interesting avenue for future work.

For this experiment, the knowledge base quality metric, $Q_{kb}$ will be the dependent variable and will be used to discern any differences in the performance of the two systems. Data analysis techniques for the this experiment are described in section 3.6.
3.5.1.2 Possible Outcomes

There are three possible conclusions that will be drawn from this experiment:

1. The two knowledge systems are indistinguishable from each other in terms of knowledge base quality.

2. On average the Prometheus knowledge system results in a higher quality knowledge base than the traditional wiki system.

3. On average the standard wiki knowledge system results in a higher quality knowledge base than the Prometheus system.

3.5.2 Experiment 2: Knowledge System Performance in a Low Usage Environment

This experiment is designed to test hypothesis 2:

"Prometheus’ threshold system raises the barrier to entry and slows the knowledge acquisition process. Thus, in a low usage scenario the threshold system may prevent Prometheus from acquiring knowledge that would otherwise be deemed acceptable. In this environment the traditional wiki system may have an equal or higher average quality of knowledge than the Prometheus system that was exposed to the same environment."

3.5.2.1 Methodology

Each system will be run in an identical operational environment over 50 runs of the simulation. The simulation parameter settings for this experiment are listed in figure 3.10. The vast majority of the parameter values for this experiment are identical to those used in experiment 1: For motivations behind parameter selections not outlined below please refer to section 3.5.1.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Wiki System</th>
<th>Custom System</th>
</tr>
</thead>
<tbody>
<tr>
<td>num_users</td>
<td>1000</td>
<td>1000</td>
</tr>
<tr>
<td>num_pages</td>
<td>10000</td>
<td>10000</td>
</tr>
<tr>
<td>num_visits</td>
<td>1000</td>
<td>1000</td>
</tr>
<tr>
<td>num_reps</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>bias</td>
<td>1.025</td>
<td>1.025</td>
</tr>
<tr>
<td>access_prob</td>
<td>0.4</td>
<td>0.4</td>
</tr>
<tr>
<td>propose_prob</td>
<td>0.25</td>
<td>0.25</td>
</tr>
<tr>
<td>judge_prob</td>
<td>0.35</td>
<td>0.35</td>
</tr>
<tr>
<td>t_add</td>
<td>0.6</td>
<td>0.6</td>
</tr>
<tr>
<td>add_m</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>add_b</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>t_add_l</td>
<td>0.3</td>
<td>0.3</td>
</tr>
<tr>
<td>add_l_m</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>add_l_b</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>t_del</td>
<td>0.4</td>
<td>0.4</td>
</tr>
<tr>
<td>del_m</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>del_b</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>user_page_beta_alpha</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>user_page_beta_beta</td>
<td>2.25</td>
<td>2.25</td>
</tr>
<tr>
<td>(wiki) del_prob</td>
<td>0.2</td>
<td>–</td>
</tr>
<tr>
<td>(Prometheus) threshold_upper</td>
<td>–</td>
<td>5</td>
</tr>
<tr>
<td>(Prometheus) threshold_lower</td>
<td>–</td>
<td>5</td>
</tr>
</tbody>
</table>

Figure 3.10: Simulation parameter settings for experiment 2.

The parameter settings of note in this experiment are the number of users and the number of visits that will take place in each run of the simulation. In this environment, each of the 1000 users will only use the system once for a total of 1000 visit events in each iteration of the simulation. This environment represents a very low usage environment wherein the threshold based knowledge acquisition system of Prometheus may hinder its effectiveness.

For this experiment, the knowledge base quality metric, $Q_{kb}$ will be the dependent variable and will be used to discern any differences in the performance of the two systems. Data analysis techniques for the this experiment are described in section 3.6.

### 3.5.2.2 Possible Outcomes

There are three possible conclusions that will be drawn from this experiment:

1. The two knowledge systems are indistinguishable from each other in terms of knowl-
2. On average the Prometheus knowledge system results in a higher quality knowledge base than the traditional wiki system when used in a low usage environment.

3. On average the standard wiki knowledge system results in a higher quality knowledge base than the Prometheus system when used in a low usage environment.

3.5.3 Experiment 3: Knowledge System Performance in High a User Interaction Environment

This experiment is designed to test hypothesis 3:

*The threshold based knowledge acquisition system of Prometheus relies heavily on user feedback in order to accept high quality knowledge and to reject low quality knowledge. Thus, in a typical usage environment the Prometheus system may perform better if users are more likely to contribute to the system by submitting changes to the knowledge base and judging the changes that other users have submitted to the knowledge base.*

3.5.3.1 Methodology

In this experiment the two knowledge systems will be exposed to the same typical usage environment which is described by the parameter settings listed in 3.11. Each system will be exposed to 50 runs in this environment. In this experiment, the typical usage environment will be the same as the one used in experiment 1, except for the probabilities for the access, propose and judge actions. Since Prometheus’s hypothesized problems occur from a lack of interaction with the system, the probability for users to take the access only branch of the visit event will be much lower at only 10%. In contrast to this, the probabilities to propose a change and to judge a change in the knowledge base will be raised to 35% and 55% respectively. These probabilities, while unlikely in a mature knowledge base, may be more representative of a knowledge base that is young and still in its initial growth phase. This
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Wiki System</th>
<th>Custom System</th>
</tr>
</thead>
<tbody>
<tr>
<td>num_users</td>
<td>1000</td>
<td>1000</td>
</tr>
<tr>
<td>num_pages</td>
<td>10000</td>
<td>10000</td>
</tr>
<tr>
<td>num_visits</td>
<td>10000</td>
<td>10000</td>
</tr>
<tr>
<td>num_reps</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>bias</td>
<td>1.025</td>
<td>1.025</td>
</tr>
<tr>
<td>access_prob</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>propose_prob</td>
<td>0.35</td>
<td>0.35</td>
</tr>
<tr>
<td>judge_prob</td>
<td>0.55</td>
<td>0.55</td>
</tr>
<tr>
<td>t_add</td>
<td>0.6</td>
<td>0.6</td>
</tr>
<tr>
<td>add_m</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>add_b</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>t_add_l</td>
<td>0.3</td>
<td>0.3</td>
</tr>
<tr>
<td>add_l_m</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>add_l_b</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>t_del</td>
<td>0.4</td>
<td>0.4</td>
</tr>
<tr>
<td>del_m</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>del_b</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>user_page_beta_alpha</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>user_page_beta_beta</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>(wiki) del_prob</td>
<td>0.2</td>
<td>–</td>
</tr>
<tr>
<td>(Prometheus) threshold_upper</td>
<td>–</td>
<td>5</td>
</tr>
<tr>
<td>(Prometheus) threshold_lower</td>
<td>–</td>
<td>5</td>
</tr>
</tbody>
</table>

Figure 3.11: Simulation parameter settings for experiment 3.

An increase in user interaction may allow Prometheus to overcome the cost of the threshold system.

For this experiment, the knowledge base quality metric, $Q_{kb}$ will be the dependent variable and will be used to discern any differences in the performance of the two systems. In addition to the quality metric this experiment will also consider the number of changes that were accepted in to the knowledge base in each iteration of the simulation. Data analysis techniques for the this experiment are described in section 3.6.

In addition to comparing the Prometheus system against the wiki system in the same environment, the results from the two knowledge systems will also be compared against their corresponding results experiment 1. These systems was exposed to an identical environment, with the exception of the user action probabilities which are much more modestly selected. This comparison will highlight the effect of the increased system interaction on the performance of the knowledge systems.
3.5.3.2  Possible Outcomes

There are three possible outcomes from this experiment:

For each knowledge system:

1. The added user interaction with the system does not make a significant difference in
   the quality of the knowledge base.

2. The increased level of user interaction with the systems has a negative effect and
   results in a lower quality knowledge base.

3. The increased level of user interaction with the systems has a positive effect and results
   in a higher quality knowledge base.

3.6  Data Analysis

Data analysis will be performed using the R language (38). In each experiment the data
Analysis will be performed using the following methodology:

For tests comparing two samples the following methodology will be used. First, the
Anderson-Darling test for normality will be used to determine if the data from each system
is normally distributed. A threshold value of 0.05 will be used to determine the result of
the test. Thus, if the p-value of the test is less than this value, the null hypothesis will
be rejected and the data will be treated as non-normal. This test was chosen because it is
widely regarded as one of the most powerful tests for normality.

Once the normality of the data has been determined a test will be used to test for
statistical significance between the differences in the means between the two data sets. If
both sets of data are normally distributed a standard two sample t-test will be used. If both
data sets are not normally distributed a non-parametric Mann-Whitney-Wilcoxon test will
be used in place of the t-test. In either case of the tests for significance between means a
p-value < 0.05 will be required to reject the null hypothesis.
For experiments comparing more than two samples it’s important to minimize the occurrence of Type 1 errors which can result in the incorrect rejection of the null hypothesis, also known as a false positive. Type 1 errors typically result when a low probability event is observed due to the number of tests being performed. For example it may be improbable for 10 tosses of a fair coin to result in 10 consecutive heads however if the test is repeated enough times the the event will eventually occur. If this effect is not considered, this could incorrectly lead to the rejection of the null hypothesis and the conclusion that the coin is not fair. In order to avoid these errors, tests that account for these errors must be used.

In tests comparing more than two normally distributed samples a one way analysis of variance (ANOVA) test will be used to determine if there is reason to continue testing for further differences. In this test, the null hypothesis states that the samples are drawn from populations with identical means. If the ANOVA test results in a rejection of this null hypothesis the student’s t-test with a Bonferroni post-hoc correction will be used to conduct comparisons between all possible pairs of means.

In order to compare multiple non-normal samples the non-parametric Kruskal - Wallis variant of the one-way ANOVA will be used to test for differences between the sample populations. As with the ANOVA test the null hypothesis states that the samples are drawn from populations with identical means. If the Kruskal - Wallis test results in a rejection of the null hypothesis then the non-parametric Mann-Whitney-Wilcoxon rank sum test with a Bonferroni post-hoc correction will be used for comparisons between samples.
Chapter 4

Results

This section presents the results obtained from the experiments described in Chapter 3, Methodology.

4.1 Experiment 1: Knowledge System Performance in a Typical Usage Environment

Experiment 1 aims to compare the performance of the traditional wiki system against the performance of the Prometheus system when both systems are exposed to a typical usage environment. The primary metric used to make this determination is the average quality of the knowledge base, $Q_{kb}$.

Figures 4.1 and 4.2 present scatter plots of the 50 values for average knowledge base quality metric for each knowledge system. Histograms of the quality metric are presented below in figures 4.3 and 4.4. Comparison between the scatter plots and histograms from the two knowledge systems indicate that there may be discernible differences in average knowledge base quality, although there is a significant amount of overlap between the two data sets.
Table 4.1 presents statistics from the analysis of the average knowledge base quality metric for each of the systems. Again, these statistics suggest that the modifications of the Prometheus design have a significant impact on the resultant quality metric. Comparison between the averages indicate that Prometheus may have performed slightly better with a mean of 0.2337 than the traditional wiki system which exhibited a mean value of 0.2214.

Figure 4.1: Wiki System: Average knowledge base quality over 50 runs.  
Figure 4.2: Prometheus: Average knowledge base quality over 50 runs.

Figure 4.3: Wiki System: Histogram of average knowledge base quality over 50 runs.  
Figure 4.4: Prometheus: Histogram of average knowledge base quality over 50 runs.

Initial results indicate that there may be a difference between the two systems however this can not be confidently stated without further statistical analysis. For data that is
Table 4.1: Initial statistics drawn from the values of the average knowledge base quality metric for the knowledge systems

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Wiki</th>
<th>Prometheus</th>
<th>Prometheus - Wiki</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum Value</td>
<td>0.1542</td>
<td>0.1922</td>
<td>0.038</td>
</tr>
<tr>
<td>First Quartile</td>
<td>0.1908</td>
<td>0.2141</td>
<td>0.0233</td>
</tr>
<tr>
<td>Median</td>
<td>0.2094</td>
<td>0.2278</td>
<td>0.0184</td>
</tr>
<tr>
<td>Mean</td>
<td>0.2214</td>
<td>0.2337</td>
<td>0.0123</td>
</tr>
<tr>
<td>Third Quartile</td>
<td>0.2500</td>
<td>0.2417</td>
<td>−0.0083</td>
</tr>
<tr>
<td>Maximum Value</td>
<td>0.3082</td>
<td>0.3557</td>
<td>0.0475</td>
</tr>
</tbody>
</table>

normally distributed, a two sample students t-test will determine this while a Wilcoxon signed rank test will do the same for data that is classified as non-normal.

Initial impressions from the shape of the histograms presented in figures 4.1 and 4.4 indicate that both sets of data are not normally distributed. In order to confirm this the Anderson-Darling test for normality can be used. For this test, the null hypothesis is that the test data is normally distributed. As stated in Chapter 3 rejection of the null hypothesis requires a p-value less than 0.05.

The Anderson-Darling test results for quality metric values obtained from the traditional wiki system provided a test statistic value of 0.9859 with a p-value of 0.01221. Thus, there is sufficient evidence to reject the null hypothesis and so the data is classified as non-normal. The tests of the quality metric from the Prometheus system resulted in a test statistic of 1.6742 with a p-value of 0.0002335. Again the null hypothesis rejected and the data is assumed to be non-normally distributed.

Since both data sets are not normally distributed a Wilcoxon rank sum test will be used to determine if there is a statistically significant difference between means. The begins with the null hypothesis that the two data sets are from the same population. Using the quality metric data from the two knowledge systems the test produces a p-value of 0.02574. From this result, we can reject the null hypothesis and can conclude that the two samples are drawn from populations that are significantly different from one another.

The results of the two sample t-test support the initial impressions of the analysis presented in 4.1. With a higher average knowledge base quality we can conclude that
<table>
<thead>
<tr>
<th>Statistic</th>
<th>Wiki</th>
<th>Prometheus</th>
<th>Prometheus - Wiki</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum Value</td>
<td>0.08196</td>
<td>0.1338</td>
<td>0.05184</td>
</tr>
<tr>
<td>First Quartile</td>
<td>0.12202</td>
<td>0.1716</td>
<td>0.04958</td>
</tr>
<tr>
<td>Median</td>
<td>0.14898</td>
<td>0.1870</td>
<td>0.03802</td>
</tr>
<tr>
<td>Mean</td>
<td>0.15746</td>
<td>0.1899</td>
<td>0.03244</td>
</tr>
<tr>
<td>Third Quartile</td>
<td>0.18132</td>
<td>0.2044</td>
<td>0.02308</td>
</tr>
<tr>
<td>Maximum Value</td>
<td>0.33606</td>
<td>0.2834</td>
<td>0.05266</td>
</tr>
</tbody>
</table>

Table 4.2: Initial statistics drawn from the values of the average knowledge base quality metric for the knowledge systems along with the difference obtained by subtracting the wiki value from the corresponding Prometheus value.

in the simulation environment used to represent a typical usage scenario the Prometheus knowledge system results in a higher quality knowledge base on average than the traditional wiki system.

4.2 Experiment 2: Knowledge System Performance in a Low Usage Environment

The objective of experiment 2 is to compare the performance of the Prometheus knowledge system and the Wiki knowledge system in an environment which can be classified as low usage. Again, the primary metric used to determine the outcome of the experiment will be the average quality of the knowledge base, $Q_{kb}$. Additionally, the volume of knowledge acquired will be considered since there is a concern that the threshold system of the Prometheus design may cause it to reject a very high proportion of the knowledge.

The initial statistics drawn from the quality metric, presented in table 4.2, show that the Prometheus data is consistently higher than the wiki data. However, this difference is small and thus may not be significant. Further investigation will be required to determine if the results indicate a significant difference in the performance of the two systems.

Much like the graphs from Experiment 1, there is a high degree of overlap between the graphs from this experiment. Scatter for the wiki and Prometheus systems are presented in figures 4.5 and 4.6 respectively. Both scatter plots show that the data points fall in the range [0.1, 0.25], though the Prometheus system seems to have a smaller number of data
points in the lower range.

This impression is confirmed in the histograms presented in figures fig:Exp2-Wiki-Quality-Histogram and 4.8. The histogram of the Prometheus data clearly demonstrates that the Prometheus system produces less runs with average quality below 0.15.

Figure 4.5: Wiki System: Average knowledge base quality over 50 runs.

Figure 4.6: Prometheus: Average knowledge base quality over 50 runs.

Anderson-Darling test for normality classifies the data from both data sets as normal. The test for the wiki data returned a p-value of 0.09858 while the Prometheus data returned...
a p-value of 0.1092. Since both values are above the 0.05 threshold the null hypotheses cannot be rejected and both sets of data are classified as normal.

Since both data sets are normally distributed, a two sample t-test was used to test for significant differences between the means. The test resulted in a p-value of 0.0001068 is small enough to result in a rejection of the null hypothesis at a 5% confidence level. Thus, the samples are determined to originate from populations with non-equivalent means.

While the Prometheus system has a slight edge from a quality perspective this may not be true in all respects. One concern with the Prometheus system is that the low usage environment prevents quality knowledge from being accepted into the system because of the barrier that is imposed by the threshold system. Figures 4.9 and 4.10 present scatter plots of the average quantity of changes that were accepted into the knowledge base for each system.

![Figure 4.9: Wiki System: Number of changes accepted into the knowledge base over 50 runs.](image)

![Figure 4.10: Prometheus: Number of changes accepted into the knowledge base over 50 runs.](image)

Table 4.3 presents a summary the data from the two knowledge systems. These statistics make it clear that the threshold based acceptance-rejection system of Prometheus isn’t superior in all respects. From a average knowledge base quality perspective the two systems are indistinguishable, however the wiki system collects a significantly higher volume of knowledge than the Prometheus system.
<table>
<thead>
<tr>
<th>Statistic</th>
<th>Wiki</th>
<th>Prometheus</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum Value</td>
<td>221.0</td>
<td>11.0</td>
</tr>
<tr>
<td>First Quartile</td>
<td>241.2</td>
<td>18.0</td>
</tr>
<tr>
<td>Median</td>
<td>248.0</td>
<td>20.0</td>
</tr>
<tr>
<td>Mean</td>
<td>248.8</td>
<td>20.7</td>
</tr>
<tr>
<td>Third Quartile</td>
<td>256.8</td>
<td>22.0</td>
</tr>
<tr>
<td>Maximum Value</td>
<td>275.0</td>
<td>31.0</td>
</tr>
</tbody>
</table>

Table 4.3: Statistical summary of the data of the number of accepted changes for each knowledge system.

4.3 Experiment 3: Knowledge System Performance in High a User Interaction Environment

The objective of experiment 3 is to compare the Prometheus knowledge system against the standard wiki system in an environment in which users are far more likely to interact with the system through change proposal and judgement than in the typical usage environment that was described in experiment 1. Again, the primary metric of success in this experiment is the average quality of the knowledge base, $Q_{kb}$. Additionally, the percentage of proposed changes that were accepted will also be used as a metric to compare system performance.

Figures 4.11 and 4.12 present the quality metric data collected from the results of the two systems arranged as scatter plots. Initially, the two scatter plots seem fairly similar in appearance with most data points falling below 0.3 in each system.

Histograms of the quality data are presented in figures 4.13 and 4.14. With the exception of a single outlier for the Wiki system both histograms seem quite similar.

Table 4.4 presents high level statistical values drawn from the quality metric data for each system. The third column presents the difference obtained by subtracting the wiki value from the corresponding Prometheus value. These values demonstrates that the results of the Prometheus system are right shifted relative to their counterparts from the traditional wiki system, although only by a small amount.

An Anderson-Darling normality test was used to determine whether the two data sets can be classified as normally distributed. The result of this test determines which method
Figure 4.11: Wiki System: Average knowledge base quality over 50 runs.

Figure 4.12: Prometheus: Average knowledge base quality over 50 runs.

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Wiki</th>
<th>Prometheus</th>
<th>Prometheus - Wiki</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum Value</td>
<td>0.1593</td>
<td>0.1970</td>
<td>0.0377</td>
</tr>
<tr>
<td>First Quartile</td>
<td>0.2036</td>
<td>0.2207</td>
<td>0.0171</td>
</tr>
<tr>
<td>Median</td>
<td>0.2175</td>
<td>0.2402</td>
<td>0.0227</td>
</tr>
<tr>
<td>Mean</td>
<td>0.2287</td>
<td>0.2412</td>
<td>0.0125</td>
</tr>
<tr>
<td>Third Quartile</td>
<td>0.2509</td>
<td>0.2552</td>
<td>0.0043</td>
</tr>
<tr>
<td>Maximum Value</td>
<td>0.4798</td>
<td>0.3398</td>
<td>0.14</td>
</tr>
</tbody>
</table>

Table 4.4: Statistical summary of the average knowledge base quality metric data for each knowledge system.

of comparison is used to test the two sample for significant differences in the means. The null hypothesis for this test is that the data is normally distributed and requires a p-value less than or equal to 0.5 for rejection. The results of the Anderson-Darling test on the wiki and Prometheus quality data were 0.0001629 and 0.1176 respectively. Thus, the Anderson-Darling test has determined that the wiki data cannot be classified as normally distributed.

Since both samples are not normally distributed, a Wilcoxon rank sum test was used to test for significant differences between the two population means. The null hypothesis for this test is that the two population means are equal. The test produced a p-value of 0.008033 which indicates that the null hypothesis can be rejected and that the two samples are not drawn from populations with equal means.

These results are consistent with the results obtained in experiment 1. In order to
Table 4.5: Knowledge quality statistics from experiments 1 and 3 for both the wiki and Prometheus knowledge systems.

determine the effect of the change in the level of user interaction the results from this experiment will be compared against those from experiment 1 which used an environment that was identical in all respects save the levels of user interaction.

Table 4.5 presents the quality metric data statistics drawn from the results of experiment 1 along side the results from experiment 3. Comparison between the results for each knowledge system in experiment 1 and experiment 3 reveal that the knowledge systems performed very similarly in the two environments from a quality perspective.

Before continuing on with pair-wise comparisons between each sample set, a one way Kruskal-Wallis (KW) test was used to determine if any of the samples are significantly different from one another. The null hypothesis of this test is that the samples are all drawn from populations with equal means. Performing the KW test resulted in a p-value
of 0.003186. Thus, we can feel confident in the rejection of the null hypothesis and the conclusion that the samples are drawn from populations with different means.

More rigid statistical analysis confirms the similarity of the wiki from experiment 3 against its counterpart from experiment 1. An non-parametric Wilcoxon rank sum test with Bonferroni post-hoc correction comparing the quality data from the wiki system in experiments 1 and 3 provides a p-value of 0.4503 which is far too high above the 0.05 threshold required to reject the null hypothesis. Thus, the samples from the wiki system in experiments 1 and 3 are not significantly different from each other and we can determine that the change in the level of interaction had no effect on the quality of the knowledge base of the wiki system in this typical usage environment.

Similar results were obtained when comparing the results of the two Prometheus systems. Again, a non-parametric Wilcoxon rank sum with Bonferroni post-hoc correction test was performed and returned a p-value of 0.18084. Again this value exceeds the limit of 0.05 that is required for statistical significance. Thus, the increased interaction with the Prometheus knowledge system did not a significant effect on the average knowledge base quality in this typical usage environment.

While the increased interaction with the Prometheus knowledge systems did not have an effect from a quality standpoint it may have had an effect on the volume of changes that were accepted into the knowledge base. In order to determine this, the same analysis techniques will be applied to the accepted change data from the two experiments.

In order to normalize the data across the two experiments, the accepted change data that will be used from each run is the percentage of changes accepted out of all changes proposed to the knowledge base. This will allow better comparison between the results from experiment 1 and experiment 3. Figures 4.15 and 4.16 present scatter plots of the data for the percentage of accepted changes for the Prometheus system across the two experiments. Both graphs seem fairly similar with the bulk of the points falling between 0.10 and 0.20.

Figures 4.17 and 4.18 present histograms of the accepted change data from experiments 1 and 3 respectively. While the two histograms have similarities, the histogram from exper-
Table 4.6: Accepted change statistics from experiments 1 and 3 for the Prometheus knowledge system.

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Prometheus (Exp 1)</th>
<th>Prometheus (Exp 3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum Value</td>
<td>0.07991</td>
<td>0.06881</td>
</tr>
<tr>
<td>First Quartile</td>
<td>0.09888</td>
<td>0.08556</td>
</tr>
<tr>
<td>Median</td>
<td>0.10435</td>
<td>0.09224</td>
</tr>
<tr>
<td>Mean</td>
<td>0.10598</td>
<td>0.09263</td>
</tr>
<tr>
<td>Third Quartile</td>
<td>0.11319</td>
<td>0.10122</td>
</tr>
<tr>
<td>Maximum Value</td>
<td>0.13943</td>
<td>0.11394</td>
</tr>
</tbody>
</table>

Experiment 1 has a much more pronounced peak whereas the histogram from experiment 3 has a less pronounced peak and a larger variance.

Table 4.6 presents the initial statistics drawn from the accepted change data for each system. Despite the seemingly significant changes in the user action probabilities, the two data sets show very similar characteristics.

Normality test on both sets of data from the Prometheus system confirmed initial suspicions that the data is normally distributed. Again, the Anderson-Darling normality test was used on each set of data, returning p-values of 0.1281 for the sample from experiment 1 and 0.6732 for the sample from experiment 3.

Since each set of data has been classified as normal an independent two sample t-test can
be used to test for significant differences in between the population means of the two samples. This test returned a p-value of 1.138e-08, which is low enough to justify rejection of the null hypothesis. Thus, we can conclude that there is significant difference between the two samples and that the change in user interaction had a significant effect on the Prometheus knowledge system from a change acceptance perspective. The changed interaction with the system caused the Prometheus system in experiment 3 to accept a smaller percentage of its proposed changes. The higher proportion of judge events seems to have allowed the system to more effectively reject low quality knowledge from the system.

4.4 Discussion of Results

The experimental results presented in chapter 4 have demonstrated that in some scenarios alternative knowledge acquisition techniques are can have a positive effect on the emergent quality of the knowledge base. This section discusses in further detail some of the questions and conclusions raised by the results.

The experiments conducted as part of this project were exploratory in nature and were intended to determine the viability of alternative knowledge acquisition strategies
in knowledge systems. Models of two systems were used in these simulation based experiments wherein knowledge systems were exposed to environments, monitored, and compared against one another.

The first model used in the experiments was a model of a traditional wiki system which has become a popular crowd sourced solution to the knowledge management problem. Wiki systems are widely used today both as public knowledge systems as well as for organizational knowledge systems. The key tenants of the wiki design are that the knowledge system has a low barrier to entry and allows anyone with access to the system to view and edit the knowledge base. This low barrier to entry is considered a strength by many because it allows for rapid development and maintenance of the knowledge base. However, some have expressed concerns about the reliability of knowledge bases built using this technique as the low barrier of entry of the wiki design makes the knowledge base vulnerable to low quality changes.

The second model used in this project was also based on the wiki design but deviated slightly from this design by employing an alternative strategy for knowledge acquisition. This alternate strategy is designed to act as a filter that prevents low quality changes from being accepted into the knowledge base. Rather than permitting all modifications to the knowledge base the Prometheus knowledge system utilizes a threshold based system for acceptance and rejection of proposed changes to the knowledge base. When changes are proposed to the knowledge base users of the knowledge system have an opportunity to vote on the changes. User votes can either be positive, in support of acceptance, or negative, in opposition to acceptance. This threshold system still allows everyone to participate in the knowledge management process but is intended to function as a mechanism through which low quality or faulty changes are rejected and high quality changes are accepted.

In each experiment the knowledge system knowledge base was blank upon each iteration. Thus, each iteration of the simulation was independent of the others. The systems were each subjected to 50 runs of the simulation framework. As each iteration of the framework executed statistics were collected and were used as data points for knowledge system
The first experiment tested the performance of the Prometheus system against the traditional wiki system in a simulated environment that represents a typical usage environment for a knowledge system. This typical usage environment is characterized as an environment in which both knowledge systems have sufficient usage to allow the development of a mature, established knowledge base. In this environment it is hypothesized that the threshold based knowledge acquisition strategy should result in a higher quality knowledge base on average than a wiki system in the same environment.

After analysis of the data from the 50 runs of each system it became evident that the average knowledge base quality from the Prometheus system was higher than the average knowledge base quality metric data gathered from the wiki system operating in the same environment. While these results were determined to be statistically significant, the question of cost versus benefit must be considered.

On average the Prometheus knowledge system in accepted just 10.6% of the changes that were proposed to the knowledge base. The rest of the changes were either rejected outright or did not gain enough support to be accepted by the end of the simulation run. This means that only one change out of every 10 changes proposed was accepted. This value seems extreme when considered relative to the traditional wiki system, which by definition accepts 100% of all changes proposed. Thus, the growth rate of the Prometheus knowledge base in this environment is approximately \( \frac{1}{10} \) of the growth rate observed in the traditional wiki system. These results indicate that the Prometheus knowledge system may not be the best choice if the rapid growth of the knowledge base is of primary concern. Although the quality of the knowledge base was higher on average, this quality comes at a significant cost in terms of quantity and the speed of the knowledge base growth.

The second experiment compared the two knowledge systems in an environment that can be classified as 'low usage'. In this environment, the two systems had a much lower volume of interaction than the systems in experiment 1. The environment used in this experiment was designed to challenge the Prometheus knowledge system by starving it of
the user interactions that it requires to accept or reject proposed changes.

In terms of the knowledge base quality metric, this experiment yielded results that were consistent with those observed in experiment 1. In terms of quality the Prometheus system produced a higher quality knowledge base on average. While the data from each system seemed similar an independent two sample t-test determined that the two samples of quality metric data were significantly different from one another.

While the Prometheus system was superior from a quality standpoint the wiki system demonstrated its strength in low usage environments when the quality metric data is considered in conjunction with the data recording the number of changes accepted into the knowledge base from each run. While the average quality is slightly lower, it’s clear that the wiki system outperforms the Prometheus system in terms of the quantity of the knowledge collected. On average, the Prometheus knowledge system only accepted just 20 changes while the wiki system collected an average of 249 changes. This is a significant distinction between the two systems and indicates the wiki system may be better suited for use in environments that have constraints that affect the number of users who interact with the knowledge system and quantity is an important consideration.

Finally, the third experiment was designed to test the hypothesis that the Prometheus system would perform better in an environment where there are a much higher proportion of users proposing changes and judging changes. The JPD that controls the user action decision, which determines how the user will use the system in each visit, was modified so that only 10% of users will access the page without performing some sort of operation that affects the knowledge base. The remaining users were split between the propose and judge actions, with 35% of the users proposing changes and 55% of the users judging changes. Like experiment 1 the knowledge base quality metric and the percent of proposed changes that were accepted into the knowledge base were used as points of comparison between the two knowledge systems.

The significant change in the operational environment, both knowledge systems performed in a similar manner as they did in experiment 1. Again, the Prometheus system
outperformed the wiki system in terms of knowledge base quality with an average quality that was higher than the wiki system.

When comparing the performance of the Prometheus system against its counterpart from experiment 1 the Wilcoxon rank sum test indicated that there was a significant difference between the quality data from experiment 1 and experiment 3.

Surprisingly, when comparing the performance of the Prometheus system against its counterpart from experiment 1 the independent Wilcoxon rank sum test indicated that there was no significant difference between the quality data from experiment 1 and experiment 3. This upper ceiling of quality may be an indication that the environment used in this experiment does not use enough variance when generating the properties of changes. Thus, the knowledge acquisition methodology alone can prevent poor quality knowledge from being accepted into the knowledge base but it cannot improve the quality of the knowledge generated by the user base.

Finally, analysis of the percentage of accepted changes between the Prometheus systems of experiment 1 and experiment 3 indicated that the modified proportions of interactions with the system caused the Prometheus system in experiment 3 to accept fewer changes. When considered in the context of the increased quality of the knowledge base this may be an indication that the changed interaction levels allowed the knowledge system to more effectively reject low quality changes.

### 4.4.1 Prometheus Design and Threshold Parameter Estimation

Each experiment described in this project pitted the traditional wiki system against the Prometheus system with threshold values of 5 for acceptance and -5 for rejection. These threshold values were chosen as best guess values based on exploratory work that ran multiple instances of the Prometheus system in a consistent environment, each time increasing the magnitude threshold value by one. Eventually, the values of -5 and 5 were chosen as a best guess for a reasonable value that produced a good quality knowledge base.
The problem with this threshold selection technique is that the thresholds were chosen in one environment and assumed to be appropriate choices in other environments as well. As was clearly demonstrated in the low usage environment, this assumption is not always appropriate. To address this, alternative parameter selection strategies could be evaluated to determine a value selection technique that is better suited for use in a generalized environment. A few possible strategies are outlined below:

There appears to be a relationship between the amount of use that the system experiences and appropriate threshold values. This presents several challenges. First, the amount of use that the system will experience is likely not a constant throughout its lifespan. As the needs of the user base evolve the system usage change over time. Second, a priori parameter selection is a challenge as it is difficult to predict system usage in uncertain domains where the user base is unrestricted and system use is voluntary. Rather than choose parameter values in advance of system use, a strategy of dynamic threshold selection may be a more appropriate methodology.

For example, threshold values based on the size of the active user base would allow the threshold values to dynamically change over time as the knowledge system becomes more popular. This would help to eliminate scenarios like those in experiment 2 where there are simply not enough users actively participating in the management of the knowledge base to support or oppose changes.

Additionally, there is no reason why the threshold values should be restricted to a single upper and lower value for the entire knowledge base. An alternate approach would be a knowledge system in which the threshold values vary from page to page. An example of this type of strategy may be a system in which the threshold value selection could be based on the popularity of the page. By using a metric such as the average number of hits in a given time frame, the knowledge system could dynamically adjust the threshold values based on the popularity of a given page. This would allow the system to adjust the thresholds relative to the amount of impact that a proposed change could have.

Another approach is to replace the current threshold system with a threshold system that
is asymmetrical. This strategy could be combined with others and would allow knowledge
base administrators to create a system that is biased toward or against knowledge addition
or rejection. These strategies may prove to be a more effective mechanism for the creation
of a high quality knowledge base.

Other possible solutions to the changing usage problem rely on modifications to the
voting mechanism that is used to prompt the system to accept or reject a piece of knowledge.
Currently, all users are considered equal from the perspective of the knowledge system.
Another approach to this may be a system in which users have an associated reputation
record which is linked to the users interactions with the knowledge system. As users propose
changes that are accepted into the knowledge base the user’s reputation improve to reflect
the user’s positive interactions with the system. Similarly, if the user proposes changes that
are rejected the user’s reputation would decrease to reflect the user’s negative interactions.
This mechanism could be used as a weighting factor when the user is expressing their opinion
about a piece of knowledge of page in the system. Users with a strong positive reputation
could have the ability to vote multiple times or to have their vote be worth more than
just a single point. Conversely, users with low reputation within the system could have
their interactions limited. This reputation system would create an environment in which
interaction with the system has personal consequences for the user. These consequences
would serve to promote positive interaction and to deter negative interaction.

4.4.2 Simulation Framework Design and Parameters

The use of a simulation framework for the initial investigation into the viability of the
Prometheus design is a strong starting point. This approach allowed models of proposed
systems to be implemented quickly without the need to worry about details such as the user
interface design, performance, or scalability of the system. This serves to focus the research
on the mechanics of the knowledge systems themselves and to eliminate a large number
of outside factors that could cause differences between the performances of the knowledge
systems.
Further, the simulation framework eliminates the need for human subjects and grants complete control over the environment in which the knowledge systems are operating. To achieve a similar level of control using real world systems and users would have been a significant task and would likely have been an intractable task within the constraints of this project.

However, the environments used within this project are at best estimations of their real-world counterparts. Thus, it’s difficult to make predictions about the performances of actual implementations of the models in a real world scenario. Further research and access to data from real world knowledge systems is required to determine whether the assumptions that have been made about these environments are appropriate. In some sense this is not of high priority for this exploratory project as the important part is that the environments used are consistent across the different knowledge systems that are being compared.

Similarly, the creation of a mapping mechanism between a real world knowledge base to a simulated knowledge base is a challenge. Conceptually it seems simple but an objective method for deterministically evaluating the truth, learnability, or conservatism of a given piece of knowledge does not exist. This is precisely the reason why crowd sourced technologies like knowledge management systems exist, to allow human users to perform knowledge centric tasks that are beyond the abilities of computers today. However, with advances in semantic systems that allows computer systems to interpret codified knowledge, a semantic knowledge system may one day be achievable.

4.4.2.1 Knowledge System Models

Whenever simulation is used the question of model validity must be raised. There is no real world implementation of the Prometheus system and so the model itself can be considered an exact match with the design of the Prometheus system.

The wiki model is not as straight forward, however. While best efforts have been taken to model the important parts of the wiki system accurately there are aspects that have not been considered. For example the discussion and revisioning capabilities of the wiki system
have not been explicitly modeled. Arguably, these components play a substantial role in
the development of the knowledge base. However since Prometheus is a modification of the
wiki design it too would benefit from the discussion and revision functionality. Because
this functionality is the same in both systems we feel justified in their exclusion as their
incorporation would add significant complexity to the framework and models but would not
serve to distinguish between the two systems.

4.4.2.2 The Simulation Framework

In addition to model verification simulation framework validation would allow researchers
to more conclusively draw parallels between the simulation results and the real world. In
its design the simulation framework makes several assumptions that could benefit from
additional research.

One example is the assumption that typical knowledge system usage scenarios can be
broken down into the three categories that are used in the framework: Access knowledge,
propose a change, and judge the knowledge. This is based on the standard Create / Read
/ Update / Delete (CRUD) design paradigm of content management systems. Despite this,
it is entirely possible that the CRUD paradigm does not accurately represent the real world
usage of operating knowledge systems.

After assuming that access / propose / judge trichotomy is valid the experiments in this
project use a JPD with values that are estimates of real world usage patterns. Experiments
1 and 2 in this project use the following probabilities for the three actions:

access 0.4
propose 0.25
judge 0.35

While experiment 3 used the following probabilities which were intended to better rep-
resent the probabilities in an environment in which the knowledge system is still in its
developmental stages:

access 0.1

propose 0.35

judge 0.55

In order to determine the validity of these assumptions access to real world knowledge system usage data, annotated according to this classification system would be required. Further, these probabilities are likely not constants through the lifespan of the knowledge system. When the knowledge base is young and still being developed knowledge acquisition is likely more probable than it is in a well established and mature knowledge base.

Currently, a piece of knowledge in the simulation framework is described by the three values for truth, learnability, and conservatism. These three properties each have an implied meaning but aren’t truly distinct from each other in implementation. Each property is handled in the same manner and the values associated with these properties don’t have any actual implications beyond their impact on the quality metric. Ideally, each one of these properties should have a unique impact on the interactions between the user and the system that more accurately reflects its role in real world knowledge management systems.

The truth property describes the relationship that exists between the codified knowledge and the real world. The user’s perception of the truth value should not be perfect but should rather be affected by the other associated properties. Primarily, learnability should have a direct influence while conservatism should have a more indirect impact.

Learnability describes how understandable the piece of knowledge is in its current representation. Thus, a user accessing a piece of knowledge that is not easily understandable should have difficulty in evaluating the truth of the knowledge. To accomplish this one possible solution would be to utilize an interaction between the user’s learnability and the knowledge learnability as a scaling factor to determine the perceived truth value.
Similarly, conservatism represents the relationship between the piece of knowledge and the rest of the knowledge in the knowledge base. If a new piece of knowledge does not agree with the rest of the knowledge in the knowledge base then that is a clear indication that something is wrong. Conservatism issues should be an indicator that the knowledge base is in need of maintenance. Thus, low conservatism values could be used to initiate a judge event on the current page.

These revisions to the way in which the simulation framework handles knowledge would more accurately model the roles that the knowledge properties play in real world KM systems.

4.4.3 Future Work

The simulation approach used in this project allows for a great deal of flexibility in terms of both the experimental and model design. While we can never be certain that the simulation is an exact representation of a real world environment it has provided a solid foundation upon which to perform exploratory research. The experiments presented in this project have demonstrated the viability of the concept that a modified knowledge acquisition technique can have a beneficial impact on the resultant knowledge base.

However this flexibility functions as a double edged sword. Because of the possibilities made available through the simulation approach, a great deal of effort was required to restrict the scope of the project to a manageable size. Thus, there is a large amount of room for further exploration within the problem space.

There is a need for further exploration into the characteristics of the Prometheus design. This project has examined the performance of the Prometheus system in three environments however there are a wide variety of other environments to consider as well. Some possible areas for further investigation include:

- Environments exhibiting different user characteristics.
• Responsive environments or environments that change over time

• Environments with different policies on knowledge and page support or opposition.

All of the experiments in this project have focused on the performance of the knowledge systems in the initial phases of the knowledge base lifespan. In order to better understand the performance of the Prometheus system it would be useful to compare the performance of two systems who are initialized with a knowledge base that is pre-populated with knowledge from an already established knowledge base. These experiments would allow researchers to gain a better understanding of the knowledge maintenance capabilities of the Prometheus design.

Real world data on the performance of the Prometheus knowledge system would be extremely useful in illustrating the effectiveness of the modified knowledge acquisition mechanism. To achieve this an actual implementation of the Prometheus design would be required. An open source wiki system could be leveraged to reduce the amount of work required to implement the system and modified. While this would still be a sizable undertaking it would be beneficial in that it would allow researchers to to fully understand the emergent effects of the Prometheus design.
Chapter 5

Conclusions

This project has presented a case for alternative knowledge acquisition strategies in knowledge management systems. It has demonstrated that the emergent characteristics of the knowledge base can be influenced by modifying the knowledge acquisition strategy employed by the knowledge system.

A modification of the standard wiki design, Prometheus, was presented which utilizes a threshold based acceptance rejection mechanism to filter poor quality knowledge before it can be added to the knowledge base. The Prometheus design represents a novel approach to wiki based knowledge management systems wherein the system itself incorporates design elements intended to improve the quality of the knowledge accrued by the system.

To evaluate the performance of Prometheus relative to a traditional wiki system a simulation framework was developed. This modular simulation framework allows researchers to easily prototype, test, and iterate on the design of knowledge systems. Further, the system allows a high degree of control over the operational environments in which the systems are compared. This allows for the rigorous testing of knowledge systems that is required to draw definitive conclusions about system performance. This simulation framework is unique in that it allows for independent, direct comparison of knowledge systems. This is a task that is difficult to achieve with real world systems and users. This capability was used to expose
the models to controlled environments and to collect metrics to evaluate the effectiveness of each knowledge system. Through this method we were able to show that there was a significant difference in the quality of knowledge collected by each knowledge system.

Three experiments were designed and run to evaluate the two knowledge systems. Each experiment was designed to test hypothesized predictions about the performance of the Prometheus system in different operational environments. Experiment 1 demonstrated that in an environment classified as a 'typical usage' environment, the average quality of the Prometheus knowledge base was significantly higher than the knowledge base of a wiki system that was exposed to the same environment. Experiment 2 exposed the knowledge systems to a low usage environment and demonstrated that the two knowledge systems were indistinguishable from each other in terms of quality, although the quantity of knowledge collected by the Prometheus system was significantly lower than that of the traditional wiki system. The third experiment tested the hypothesis that the Prometheus system would perform better if the proportion of users proposing and judging proposed additions to the knowledge base was significantly increased. For this experiment, the same typical usage environment from experiment 1 was used with the user action probabilities significantly shifted to favour the proposal and judgement actions at the expense of the 'read only' action. Despite these significant changes to the simulation environment the results of this experiment yielded no difference between the results of experiment 1 and the results of experiment 3.

Finally, some possible avenues for further work were discussed. Topics discussed included improvements of the simulation framework, a desire for stronger model validation, and suggestions of other areas of interest for further experimentation. These improvements, along side an actual implementation of the Prometheus system, would provide researchers with a solid understanding of the effects of the Prometheus design.
Bibliography


University as a Bridge from Technology to Society. IEEE International Symposium on Knowledge management and its application model in enterprise information systems, 2000, p. 287.


Appendix A

Glossary

**knowledge acquisition** *Knowledge acquisition*, often also referred to as knowledge creation, refers to the process through which knowledge is added to a knowledge base in a knowledge system. Typically, this refers to the process of taking knowledge from an expert user and adding it to the knowledge base, although this term could be applied to other means of integrating external knowledge into the knowledge base such as the use of a web crawler.

**knowledge base** A *knowledge base* is a common repository of knowledge that is made available to a group of knowledge consumers. Knowledge bases usually apply some sort of codification and storage mechanism to allow for deterministic storage and retrieval of knowledge. In electronic systems a knowledge base often takes the form of a database although other storage mechanisms exist. Other examples of knowledge bases may include a filing cabinet, notebook, or even a library.

**knowledge management** Knowledge management refers to activities and practices involved in the creation, organization, maintenance, and dissemination of pieces of knowledge contained in a knowledge base.

**knowledge management system** A knowledge management system is any system designed to support the practices of knowledge management.
**wiki**  A wiki is a web site that is openly editable by anyone. Users may freely add, edit and delete content as they see fit. Often, wikis are used collaboratively. The first wiki was created by Ward Cunningham.