Department of Computer Science
University College London

Support for Collaborative
Elaboration of Requirements
Models

Camilo Fitzgerald

Transfer VIVA

June 2009
# Table of Contents

1 Problem Statement........................................................................................................ 4

2 Literature Review........................................................................................................ 6
   2.1 Requirements Elaboration ..................................................................................... 6
   2.2 Asynchronous Requirements Elaboration .......................................................... 7
   2.3 Requirements Inspection and Critique ................................................................. 8
   2.4 Collaborative Sense-Making Technologies ......................................................... 9
   2.5 Conclusions ........................................................................................................ 9

3 Proposed Contribution ................................................................................................ 12
   3.1 Approach ........................................................................................................... 12
   3.2 Scope ................................................................................................................ 12

4 Research Progress ..................................................................................................... 14
   4.3 Conceptual Framework ..................................................................................... 14
   4.4 Issue Taxonomy ................................................................................................ 16
   4.5 Model Transforms .............................................................................................. 18
   4.6 Issue Lifecycle .................................................................................................. 20
   4.7 Validation of Framework ................................................................................... 21

5 Plan for Completion ................................................................................................... 24
   5.8 Overview ............................................................................................................ 24
   5.9 Immediate Concerns .......................................................................................... 25
   5.10 Future Enhancements ....................................................................................... 26
   5.11 Validation ......................................................................................................... 27

6 Appendix A – Envisioned Prototype ......................................................................... 29
   6.1 Developers create a goal graph ......................................................................... 29
   6.2 Annotating an assertion with an issue .............................................................. 30
   6.3 Suggesting responses to an issue ..................................................................... 31
   6.4 Suggesting a model alteration response ......................................................... 32
   6.5 Arguing over issue responses .......................................................................... 33
   6.6 Selecting an offline activity as a response ....................................................... 34

7 Appendix B – Firefox Issue Ticket Extract ............................................................... 35

8 Appendix C – Firefox example .................................................................................. 42

9 Appendix D – Thesis Table of Contents ................................................................... 49
Abstract

Requirements engineering activities are often carried out using web-based communications and tools due to their highly collaborative nature. Surprisingly, little by way of support has been developed to aid those working in collaborative environment with the requirements elaboration process.

A framework is therefore suggested to provide aid with the construction and criticism of requirements goal models in an asynchronous collaborative setting. A taxonomy of issues is offered to developers that is used to critique elements of a goal graph. Support is then given for suggesting resolutions to these issues, choosing among them, and applying them.

This report gives the details of the research to date on this framework, and a plan for its completion as part of a PhD Thesis submission.
I Problem Statement

A survey of large IT projects in the US conducted in 1995 found requirements related issues to be perceived as the major cause for impaired, challenged and failed projects (Standish Group 1995). A more recent report from the Meta Group in 2003 claimed that 60-70% of IT project failures are still to be attributed to poor requirements management. Correspondingly, a rapid growth has been seen in the past two decades in the field of requirements engineering as a response to the industry’s need for better solutions.

Software development using a collaborative online environment to house all communications and work-in-progress has proven to be a success, as projects like Mozilla’s Firefox and Ubuntu have shown. Requirements elaboration methods such as goal modelling frameworks (Axel van Lamsweerde 2001) and conflict management techniques (Robinson et al. 2003), however, are currently based around offline work and are not supported in these settings.

The reliance of existing techniques on synchronous offline work has made them difficult to use in a collaborative web-based environment. Instead, requirements activities are conducted in an informal fashion using email, forums and mailing lists in both open source development (Bird et al. 2008) and commercial organisations (Damian 2004). The 2007 requirements engineering roadmap (Cheng & Atlee 2007) identified this as a prominent issue facing the research community, stating that practitioners need techniques to facilitate and manage distributed requirements activities.

A case study we conducted on the Firefox project (Fitzgerald 2008) found that requirements elaboration activities were carried out in an issue tracker, similar to a forum. Requirements artefacts were added and critiqued by creating issue tickets, and the resulting elaboration discussions were held in chronologically ordered posts. These discussions were often undirected, unproductive and lengthy. The case study identified the key problems causing this, which may be split into two categories.

The first category concerns those problems specific to the requirements elaboration process:

- Hidden assumptions
- Ambiguities
- Incompleteness in the requirements specification
- Non-minimality of a requirements specification, resulting in irrelevant requirements
- Inconsistencies and conflicts among requirements
- Difficulties in requirements change management
- Lack of stakeholder participation
- Time and effort is needed to understand a project goals and history
- Infeasible requirements
o Non-testable requirements, which cannot be tested for satisfaction
o Over-Specification, where a requirements specification gives details of the solution instead of the requirements for the solution
o Unintelligibility

The second category contains the problems that are found when using asynchronous collaborative technologies for general-purpose discussions, which correspond with those identified in (M. Klein & Iandoli 2008):

o Lack of structure – Discussions frequently lacked structure, which resulted in digression from the topic and contributed to the other problems in this list.

o The Soapbox problem – Those who speak ‘loudest’ or more frequently make more impact, as opposed to those who may be contributing good points but may not have the voice or the time to make themselves heard.

o Flawed argumentation – A lack of structure in argumentation leads to pathological discussions, examples of which include circularity and discussants arguing when they have not fully understood each other’s points.

o Noise – The problems listed in this category, coupled other factors such as communications irrelevant to the topic lead to a high signal-to-noise ratio. This resulted in it being difficult to follow the key points of a discussion, and also meant that the barrier to entry to a discussion was heightened.

o Scattered content – As a result of the noise and lack of structure content was scattered around the environment, also making discussions hard to follow and increasing the barrier to entry.

Requirements modelling frameworks and heuristics, such as KAOS goal modelling (Landtsheer 2007), go towards addressing the problems in the former category, but do not translate to an asynchronous collaborative setting. Meanwhile, sense-making techniques and tools including Deliberatorium (M. Klein & Iandoli 2008) and Compendium (Shum et al. 2006) go some way in solving the problems in the latter category, but do not cover the requirements elaboration process. A review follows in the next section in which the body of literature relating to the problem is examined, and it is shown that new techniques are needed to solve these problems.
2 Literature Review

A review of the literature relevant to the asynchronous requirements elaboration follows in this section, split into four sub-sections. Firstly, general requirements elaboration techniques are discussed, with a view on extending them for use in collaborative environments. The next sub-section reviews the little research that does exist on elaboration in these settings. Finally, the last two sub-sections cover literature that provides insight into solving the asynchronous requirements elaboration problem, namely requirements review schemes and sense-making technologies.

2.1 Requirements Elaboration

The field of requirements engineering has seen a rapid growth in a relatively short time as a response to the industry’s need to better techniques (Curtis et al. 1988). Models are considered to be fundamental to supporting requirements engineers at different stages of the requirements process (Nuseibeh & Easterbrook 2000). Accordingly, many such modelling frameworks have been developed to help with different RE activities, ranging from stakeholder analysis (Alexander 2005) to the formal specification of requirements (Axel van Lamsweerde 2000).

A key set of activities in the RE process come under the heading of requirements elaboration, in which the body of requirements and goals is expanded through analysis, usually taking the form of reductions, abstractions and conflict resolutions. Models and techniques supporting this analysis vary in the stages of the requirements lifecycle in which they are used and the level of support they provide.

The most familiar method for conducting requirements elaboration, included in UML and most requirements engineering textbooks, involves the use of use cases, scenarios and user stories. This traditional approach, however, has many flaws such as its inability to manage conflicts or check for the completeness of a requirements specification. Several new solutions have arisen from the literature in response to these problems.

One such solution is the WinWin approach (B. Boehm et al. 1995), which supports early stage requirements negotiation. Elaborations within this framework take place by starting with an initial set of stakeholders’ win conditions, and detecting conflicts between them. These conflicts are then resolved by constructing points of agreement. This process is then iterated and further conflicts are identified conflicts and resolved. Case studies have suggested that the approach leads to improved cooperation and focus on the key issues (B. Boehm & Egyed 1998). WinWin, however, is an approach for agreeing upon high-level stakeholder goals, rather than complete systems requirements elaboration.
Techniques are therefore needed to systematically drive forward the elaboration process once these high-level goals have been discovered. KAOS (Dardenne et al. 1993) and i* (Yu 1997) are two such techniques that have received a substantial following in the literature, and are both examples of Goal Oriented Requirements Engineering (GORE) frameworks. GORE goes beyond aiding the requirements elaboration process by providing other benefits to the requirements engineer (Axel van Lamsweerde 2001):

- Requirements completeness – whereby high-level goals can shown to be satisfied by the requirements.
- Requirements pertinence – whereby irrelevant requirements that do not contribute to high-level goals are avoided.
- Communication of the requirements and their rationale to stakeholders.
- Improvement of the structure and readability of requirements specifications.
- Description and exploration of alternative refinements of requirements and high-level goals.
- Support for the management of conflicts between requirements.
- Separating stable requirements and goals from those that are less likely to change.

### 2.2 Asynchronous Requirements Elaboration

Rational DOORS is one of many examples of the tools currently available that allow developers to document and maintain requirements across sites. These tools, however, only allow for the depositing, viewing and checkout of requirements artefacts and do not provide support for the process of developing and elaborating these requirements collaboratively. The lack of support in this aspect makes them inadequate for projects where cross-site developers wish to collaborate on requirements model.

Collaborative activities are therefore handled in cross-site projects via email, forums and mailing lists (Bird et al. 2008; Damian 2004), thus losing the advantages gained from the use of requirements elaboration techniques. This problem has been not been addressed in the requirements literature, with a few recent exceptions, all of which are in early stages of development.

WikiWinWin (Yang et al. 2008) is once such exception in which wiki-based support is given to developers using the WinWin method across multiple sites. It allows stakeholders and developers to house win-conditions, points-of-agreement and other concepts in a web-based environment. The task of identifying conflicts and proposing resolutions still, however, remains an offline activity.
The SOP project (Decker et al. 2007) provides a similar web-based collaborative environment for working on scenarios and use cases. Like WikiWinWin, it does not support the process of creating these artefacts, but instead allows developers to house artefacts online. Decker does mention, however, that the project may take a future direction of utilising usage data within the wiki to aid with the process. For example, repetitive edits to wiki pages, or so-called 'flame wars', could be detected and flagged as possible conflicts between users.

2.3 Requirements Inspection and Critique

The inspection and critique of requirements models and documents is an activity that takes place in almost every software project, and is key to the elaboration phase. In many cases this is done informally, although structured and formal methods for performing this task have arisen from research to support the task. The extra effort required to communicate in an asynchronous web-based setting suggests that increased benefits would be gained from the use of structured, or formalised, requirements critique.

The most basic form of structured critique takes the form of a checklist, which a reviewer uses to check each requirement for issues. Lamsweerde (Axel van Lamsweerde 2009) has compiled a comprehensive checklist, which includes items such as verifying a requirements' feasibility and checking for inconsistencies between requirements.

Fickas' paper on critiquing software specifications (S. Fickas & Nagarajan 1988), which formed the basis of the critiques feature in ArgoUML, is an example of an automated solution. Requirements, high-level goals and use cases are linked with the help of a user. An automated critiquer then checks for high-level goals that are not satisfied in the requirements, use cases that contribute negatively to high-level goals, and use cases that do not contribute to the high-level goals.

Finkelstein has developed a schema to aid with the process of asynchronous requirements inspection between two parties though structured argumentation (Finkelstein 1992). A set of speech acts and rules governing their usage are used to create, review and correct requirements documents. This work, however, does not support developers in identifying specific requirements issues, but instead allows them to find general faults and argue about them.
2.4 Collaborative Sense-Making Technologies

The sense-making body of work, although not directly intended for requirements processes, has made substantial progress in identifying and providing solutions to the problems faced in asynchronous collaboration. The term ‘sense-making’ stems from the Intelligent Systems field and has been defined as research related to how people make sense of the world (G. Klein et al. 2006).

Compendium (Shum et al. 2006) and The MIT Collaboratorium (M. Klein & Iandoli 2008) are two examples of research projects aimed at aiding people in asynchronous environments to collaboratively build models that represent how they make sense of world issues. A requirements document as a specialised instance of such a knowledge representation model, and accordingly insight can be gained from how the sense-making community have addressed the problems faced with collaboration, communication and argumentation.

Indeed, many of the problems faced in a study of requirements management in Firefox project (Fitzgerald 2008) were similar to those that the sense-making community is trying to solve. These are chiefly problems of lack of structure, the soapbox problem, flawed argumentation, noise and scattered content (M. Klein & Iandoli 2008). A time based structure for arguments, which as also common to the Firefox case study, gives rise to many of these problems.

Sense-making solutions avoid this by structuring a debate according to an argument, which also encourages systematic and complete arguments. Different solutions, however, structure their arguments in alternative ways. In Compendium, for example, Issues, Ideas and Arguments are used to express questions, answers to a question, and statements that support or detract from an idea respectively. The various ways in which this has been solved provide insight into solutions for an asynchronous web-based requirements elaboration solution.

2.5 Conclusions

In the first sub-section on Requirements Elaboration we have discussed current elaboration techniques and the value they provide to requirements engineers. These techniques, however, cannot be applied in an asynchronous web-based environment, in which they would be of much use.

Work exists that attempts provide elaboration techniques in an asynchronous environment, such as the SOP project and WikiWinWin. These works are in very early stages and only cover use-cases, or WinWin based elaboration. Furthermore, they do not provide much by way of support for the elaboration process, but revolve around allowing users to house and
view artefacts online. The community is lacking a framework supporting use of a more complete technique, such as the goal-graphing method.

Two areas of the literature were reviewed in the last sub-sections that provide insight into possible asynchronous requirements elaboration solutions. Inspection and review methods could be used to support the process, whether automated, checklist-based, or argumentation based. The lessons learnt from the sense-making community in their use of structured argumentation are also likely to be useful to the design of such a solution.

Finkelstein’s schema for the review and correction of requirements specifications is an early attempt at supporting asynchronous requirements elaboration. It does not, however, support more than two parties. Furthermore, it may only be used to work on text-based documents, rather than supporting more complete requirements elaboration methods, thus losing the benefits gained from their usage.

In conclusion, we are lacking support for the collaborative elaboration of requirements in asynchronous web-based environments. Much work exists that can be built upon, including:

- Modelling techniques that could be adapted for an asynchronous environment, such as KAOS goal modelling.
- Sense-making technologies that have solved some of the problems with communication and collaboration.
- Inspection and review techniques that could be adapted to allow for collaborative criticism and correction of requirements artefacts.
- Existing works, such as the SOP project and WikiWinWin, which have also attempted to solve the asynchronous elaboration problem.

Table 2.1 summarises the literature reviewed in this section in terms of its ability to deal with those problems found in asynchronous requirements elaboration, as defined in Section 1.
### Modelling Support

<table>
<thead>
<tr>
<th>KAOS</th>
<th>i²</th>
<th>DOORS Wiki win-win</th>
<th>The SOP Project</th>
<th>ArgoUML</th>
<th>Finkelstein’s Schema</th>
<th>Collaboratorium / Compendium</th>
</tr>
</thead>
<tbody>
<tr>
<td>Support for goals</td>
<td>✓</td>
<td>✓</td>
<td>x</td>
<td>✓</td>
<td>x</td>
<td>n/a</td>
</tr>
<tr>
<td>Support for requirements</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>x</td>
<td>✓</td>
<td>n/a</td>
</tr>
<tr>
<td>Conceptual completeness</td>
<td>✓</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>n/a</td>
</tr>
<tr>
<td>Conceptual pertinence</td>
<td>✓</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>n/a</td>
</tr>
<tr>
<td>Support for Formal Reasoning</td>
<td>✓</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>n/a</td>
</tr>
<tr>
<td>Improved communicability of concepts</td>
<td>✓</td>
<td>✓</td>
<td>x</td>
<td>✓</td>
<td>x</td>
<td>n/a</td>
</tr>
<tr>
<td>Improved structure</td>
<td>✓</td>
<td>✓</td>
<td>x</td>
<td>✓</td>
<td>x</td>
<td>n/a</td>
</tr>
<tr>
<td>Support for alternate refinements</td>
<td>✓</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>n/a</td>
</tr>
<tr>
<td>Conflict management</td>
<td>✓</td>
<td>✓</td>
<td>x</td>
<td>✓</td>
<td>x</td>
<td>n/a</td>
</tr>
<tr>
<td>Separation of stable from non-stable concepts</td>
<td>✓</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>n/a</td>
</tr>
<tr>
<td>De-scattered content</td>
<td>✓</td>
<td>✓</td>
<td>x</td>
<td>✓</td>
<td>x</td>
<td>n/a</td>
</tr>
</tbody>
</table>

### Collaborative Support

<table>
<thead>
<tr>
<th>KAOS</th>
<th>i²</th>
<th>DOORS Wiki win-win</th>
<th>The SOP Project</th>
<th>ArgoUML</th>
<th>Finkelstein’s Schema</th>
<th>Collaboratorium / Compendium</th>
</tr>
</thead>
<tbody>
<tr>
<td>Support for cross-site collaboration</td>
<td>n/a</td>
<td>n/a</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Support for asynchronous collaboration</td>
<td>n/a</td>
<td>n/a</td>
<td>x</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Deals with soapbox problem</td>
<td>n/a</td>
<td>n/a</td>
<td>x</td>
<td>✓</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Improved argumentation</td>
<td>n/a</td>
<td>n/a</td>
<td>x</td>
<td>✓</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Noise Reduction</td>
<td>n/a</td>
<td>n/a</td>
<td>x</td>
<td>✓</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Collaborative model critique</td>
<td>n/a</td>
<td>n/a</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>✓</td>
</tr>
</tbody>
</table>

*Table 2.1 - Literature Reviewed vs. Objectives for Asynchronous Elaboration*
3 Proposed Contribution

This section outlines the approach and scope of the proposed thesis contribution, which is offered as a solution to the problem set out in previous sections.

3.1 Approach

We propose to support asynchronous requirements elaboration, a real world problem that has been largely ignored in the literature. The framework and schema described in this document combines sound requirements elaboration techniques with the benefits of asynchronous collaboration methods.

We have chosen to adapt the KAOS goal-modelling framework (Dardenne et al. 1993) to this purpose, as it encourages specifications that are complete, pertinent, easily communicable and well structured. Furthermore, it allows for alternative refinements of specifications, conflict management and the separation of stable requirements from those that are likely to change.

In order to frame KAOS in an asynchronous setting we have chosen to provide a system of critiques, which has previously been used as an effective solution to the problem (Finkelstein 1992; S. Fickas & Nagarajan 1988). This involves annotating goal model elements with issues, which are then discussed and resolved collaboratively.

Concepts taken from the sense-making community have been used to form the basis the discussion and selection of these issues. Furthermore, we have reused advances made in the community to structure the collaborative environment for facilitation and encouragement of collaboration.

3.2 Scope

The main contribution of the thesis is support for asynchronous goal-graph elaboration via the critique of goal model elements. This will remain at the core of the scope.

We have chosen, however, to provide further aid with the placement of goal model critiques. A taxonomy of issues typically found in goal-graphs is therefore provided, allowing developers to pinpoint their critiques. These issues also have resolutions strategies associated with them, which are used to advise developers on how to proceed. We believe that this will be of significant benefit, as it will drive forward the elaboration process.
We are also currently considering further extensions to the contribution, including the automatic generation of issues, and an extended argumentation system. Automatic generation of issues involves inspecting a goal graph for possible issues, resulting in automated critiques. Some of these checks could be built-in, such as cyclic paths or use of words that are known to be ambiguous, while others could be user defined. An extended argumentation system, meanwhile, would implement extended argumentation logic similar to that of Finkelstein’s schema (Finkelstein 1992) that would push forward discussions and model elaborations. Section 5 discusses further these and other possible extensions.

We have chosen only to include the basic goal modelling components of KAOS in our framework, as they are the minimum needed to work on a requirements specification. These are goals, domain assumptions, AND-Refinements, and a glossary. This will facilitate adoption of the new techniques, as there will be less of a learning curve. Furthermore, this simplifies the development of the framework, as the full richness of the KAOS language will not be dealt with. More concepts, however, may be added at a later date to provide further value, such as OR-Refinements and formal definitions, which allow for alternative refinements and more complete specifications respectively.
4 Research Progress

Research to date has culminated in the design of the framework and schema that follows from the proposed contribution outlined in the previous section. An overview of the framework is first given, followed by the details of its usage. Finally, a short example is used to demonstrate its benefits.

4.3 Conceptual Framework

An overview of the contribution’s framework design can be seen in Figure 4.1, which depicts a high-level entity-relationship diagram. The diagram is separated into two sections, with one section describing a standard goal model, and the second describing the extended annotations that may be made on the model to provide asynchronous support.

Appendix A describes an envisioned prototype that uses this framework, and demonstrates how it might be used to provide a tool for asynchronous requirements elaboration. A summary of how this framework might be used is as follows:

1. Developers collaboratively construct a goal-graph, composed of model elements.

2. If a developer believes there to be a problem with a goal model element, they annotate it with an issue.

3. Collaborators then propose actions to deal with this issue, which fall into the following three categories:
   a. They may see the issue as a problem with the requirements model, and suggest a requirements model transformation that resolves it.
   b. They may believe the issue not to be a problem and suggest removing the issue annotation by proposing to ignore the issue.
   c. If they believe that more information must be gathered before dealing with the issue they can propose an offline activity to do so.

4. Participants may then attach pros and cons to proposed actions, which justify or criticise them.

5. When moderator feels that enough proposed actions, and arguments for and against them have been suggested they then select one of these proposed actions. The model is then altered accordingly:
   a. A model transform applies the suggested transformation to the model.
   b. Choosing to ignore the issue removes the issue and makes no further changes to the model.
   c. Opting to undertake an offline activity locks the issue until the activity has been completed, at which point discussions can be re-opened.
Figure 4.1 – Framework overview
4.4 Issue Taxonomy

The schema contains a taxonomy of issues typically found in goal models that is made available to developers to direct their critiques of the requirements model. Categorisations of requirements document defects can already be found in the literature. Such a categorisation, however, does not currently exist specifically for goal-graphs. A standard list of defects has therefore been adapted from Lamsweerde’s textbook (Axel van Lamsweerde 2009), which comprehensively covers requirements documents defects.

The set of issues, shown in Table 4.2 has been sub-divided into three categories according to the model elements on which they may be placed; assertions, AND-refinements and in-node assertion text. This allows developers to identify the issue they wish to place more easily.

<table>
<thead>
<tr>
<th>Issues Placed on Assertions</th>
<th>Placement</th>
<th>Informal Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Infeasible</td>
<td>A Goal</td>
<td>Implementation of the goal is not feasible.</td>
</tr>
<tr>
<td>Not Measurable</td>
<td>A Goal</td>
<td>There is no feasible way to measure the degree to which the goal is satisfied, and therefore no way tell if the system, once built, has achieved the goal.</td>
</tr>
<tr>
<td>Logical Inconsistency</td>
<td>I – N Assertions</td>
<td>The assertions cannot coexist, either because they are logically inconsistent or contribute negatively to each other.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Issues Placed on AND-Refinement Links</th>
<th>Placement</th>
<th>Informal Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uncontributive Reduction</td>
<td>A Reduction link</td>
<td>The child of the reduction link does not contribute to the parent.</td>
</tr>
<tr>
<td>Incomplete Reduction</td>
<td>All reduction links emerging from an goal</td>
<td>Satisfying the full set of child assertions will not satisfy the parent goal.</td>
</tr>
<tr>
<td>Non-Minimal Reduction</td>
<td>All reduction links emerging from a goal</td>
<td>Not all of the child assertions need to be satisfied in order to satisfy the parent goal</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Issues Placed on In-Node Assertion Text</th>
<th>Placement</th>
<th>Informal Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ambiguous</td>
<td>A selection assertion text</td>
<td>There are multiple ways to interpret a section of text.</td>
</tr>
<tr>
<td>Irrelevant</td>
<td>A selection assertion text</td>
<td>The selected text is irrelevant to the specification.</td>
</tr>
</tbody>
</table>
### Table 4.2 – Taxonomy of Issues

The taxonomy was adapted in two stages. Firstly, by removing the defects not relevant to goal-graphs, or those whose addition to the taxonomy would provide little benefit, and secondly by the inclusion of additional defects found to be common in goal graphs.

Below follows a list of those defects from (Axel van Lamsweerde 2009) that have not been used, and the rationale behind the choices. Those defects that were renamed for the purpose of clarity are also shown:

- **Omission** – A framework for a goal graph, as opposed to that of a requirements document, is formally defined and has syntactical rules that the schema automatically enforces. This already counters most omission cases, meaning that a separate annotation for the purpose is not needed. One special type of omission may still surface when a goal has not been fully reduced. This is covered by the Incomplete Reduction annotation.

- **Contradiction** – Replaced by specific contradiction types relevant to goal graphs. Namely Designation Clashes, Terminology Clashes and Logical Inconsistencies.

- **Inadequacy** – This term covers a requirement or goal that does not accurately reflect the stakeholders’ needs. In a goal graph high-level goals are used to reflect the stakeholders’ needs, which are then reduced into further goals and requirements. In this schema we assume discussions over these high level goals to be out of scope. It may, however, prove to be useful to include an Inadequacy Annotation in the schema’s taxonomy at a later date.

- **Unmeasurability** – Replaced by the term Not Measurable for the sake of clarity.

- **Noise** – Replaced by the term Irrelevant for the sake for clarity.

- **Poor structuring** – A framework for a goal graph, as opposed to that of a requirements document, is formally defined and has syntactical rules that the schema automatically enforces. This means that problems with of poor structuring will be minimised. It may, however, prove to be useful to include a Poor Structuring Annotation in the schema’s taxonomy at a later date.

- **Forward reference** – A goal graph, unlike a requirements document, does not have an order. It is therefore not possible to forward reference.
- Remorse – This term refers to backwards referencing, which is an issue that is not applicable to goal graphs for the same reason as forward referencing.

- Poor modifiability – This is an issue sometimes raised with goal graphs, in which the propagation changes that result from the alteration of a goal are costly. This is a difficult problem to tackle and there is almost no literature supporting it. We will therefore leave it out of scope and provide no aid to developers in developing a goal graph that optimises its structure in anticipation of changes. It may, however, prove to be useful to include a Poor Modifiability Annotation in the schema’s taxonomy at a later date.

- Opacity – This term refers to dependencies that have not been made clear in a requirements document. A framework for a goal graph, as opposed to that of a requirements document, is formally defined and has syntactical rules that the schema automatically enforces. This means that opacity issues will be minimised. It may, however, prove to be useful to include an Opacity Annotation in the schema’s taxonomy at a later date.

Following a study of goal-graphing literature focused on identifying defects typically found in goal graphs, further issues were added to the taxonomy. Logical Inconsistency, Designation Clash and Terminology Clash were taken from (A van Lamsweerde et al. 1998), where a categorisation of logical inconsistencies found in goal graphs is given. Uncontributive Reduction, Incomplete Reduction and Non-Minimal Reduction were taken from (A. Van Lamsweerde et al. 1995), in which a categorisation of problems found when elaborating goal graphs is detailed.

### 4.5 Model Transforms

The schema contains rules as to which model transformations resolve particular issues that are provided as guidance to developers. Guidance on model transforms is offered, rather than enforced, to allow for process flexibility. This decision was taken as it has been found processes that are too rigid are often not adopted (Fuggetta 2000).

The set of rules for these transforms, shown in Table 4.3, has been constructed using recommendations from the literature (A van Lamsweerde et al. 1998) and through intuition. We are currently in the process of formalising the set, and sourcing it more consistently. Suggested transforms are categorised into Add, Edit and Delete transformations, one or more of which may be applied to resolve an issue.
<table>
<thead>
<tr>
<th>Issue</th>
<th>Model Element Removal</th>
<th>Edit Assertion</th>
<th>Add New Assertion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Infeasible</td>
<td>Remove the goal on which it is placed</td>
<td>Edit the in-node text of the goal to make it feasible</td>
<td>×</td>
</tr>
<tr>
<td>Not Measurable</td>
<td>Remove the goal on which it is placed</td>
<td>Edit the in-node text of the goal to make it measurable</td>
<td>×</td>
</tr>
<tr>
<td>Logical Inconsistency</td>
<td>Remove one of the assertions on which it is placed</td>
<td>Weaken one of the assertions on which it is placed</td>
<td>The new assertion(s) must serve to avoid the conflict</td>
</tr>
<tr>
<td>Uncontributive Reduction</td>
<td>Remove the reduction link</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>Incomplete Reduction</td>
<td>×</td>
<td>Strengthen the child assertions OR weaken the parent goal</td>
<td>The new assertion(s) must serve to complete the reduction</td>
</tr>
<tr>
<td>Non-Minimal Reduction</td>
<td>Remove one or more of the reduction links on which it is placed</td>
<td>Strengthen the parent goal</td>
<td>×</td>
</tr>
<tr>
<td>Ambiguous</td>
<td>Removal of the ambiguous in-node text indicated</td>
<td>Replace the in-node text with one of the options given indicated in the annotation</td>
<td>×</td>
</tr>
<tr>
<td>Irrelevant</td>
<td>Removal of the irrelevant in-node text indicated</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>Over-Specification</td>
<td>Removal of the over-specified in-node text indicated</td>
<td>Edit the in-node text to be less implementation specific</td>
<td>×</td>
</tr>
<tr>
<td>Unintelligible</td>
<td>Removal of the unintelligible in-node text indicated</td>
<td>Edit the in-node text to make it more intelligible</td>
<td>×</td>
</tr>
<tr>
<td>Designation Clash</td>
<td>Removal of all but one of indicated in-node texts</td>
<td>Re-term all but one indicated in-node texts the terms suggested for each respective occurrence in the annotation</td>
<td>×</td>
</tr>
<tr>
<td>Terminology Clash</td>
<td>Removal of all but one of indicated in-node texts</td>
<td>Re-term all indicated in-node texts with one of the terms suggested in the annotation</td>
<td>×</td>
</tr>
</tbody>
</table>

Table 4.3 – Suggested Model Transform Actions
4.6 Issue Lifecycle

Diagram 4.4 shows the lifecycle of an issue, from placement on the goal graph to removal. Appendix A contains concrete examples of how this state chart diagram might be used to construct and elaborate a goal graph in a prototype tool. Activities that may take place in each state are as follows:

- S0: Newly Placed – An issue is placed upon a goal model element. Contributors may then propose actions that attempt to resolve the issue.
- S1: In Discussion – This state is entered when the first proposed action is submitted. Contributors may then continue to propose other actions, and associate these with pros and cons.
- S2: Pending Offline Activity – If a moderator chooses to select an offline activity as a response to the issue further discussions are locked until the activity has been completed.
- S3: Resolved – When a moderator chooses to either implement a model transform, or to ignore the issue it then moves into the resolved state.

Diagram 4.4 – Issue Lifecycle
4.7 Validation of Framework

A short example is now presented that demonstrates the benefits of the conceptual framework we have presented in this section. Flaws with the requirements elaboration process in the Mozilla Firefox project are first discussed. Following this, an example is reconstructed on our framework, and it is shown how these problems are dealt with.

The example concerns the addition to the Firefox requirements document of a new requirement to include Wikipedia among the list of default searches made available in Firefox. The original discussion took place in over 200 posts, took over 2 years to come to completion and was frequently heated and undirected. It is adapted from a case study that can be found in (Fitzgerald 2008).

Appendix B contains an extract from an Issue Ticket in which requirements discussions were held, and exemplifies the typical elaboration process for the Firefox project. The process suffers from multiple problems, including those common to unstructured asynchronous discussions (M. Klein & Iandoli 2008):

- **A lack of structure** – This can be seen throughout the issue ticket, for example posts #3 & #10 concerning the current Firefox implementation are interspersed with posts #6 & #9, which house arguments for the inclusion of Wikipedia in the list of default search engines.
- **The soapbox problem** – Basil Hashem, for example, submitted 7 of the 19 posts in the extract, whereas the highest number of posts submitted by another discussant is 2. It is therefore likely that Basil’s opinions are heard above those of the other discussants.
- **Flawed argumentation** – The conversation strays from the subject in Post #15, where the argument is made to keep answers.com as a default search option because “answers.com is not Yahoo”.
- **Noise** – Post #19 gives the details of how the Wikipedia search algorithm works, which bears little relevance to the requirements elaboration.
- **Scattered content** – As a direct result of the lack of structure described above, arguments supporting the addition of Wikipedia to the default search bar are interspersed details of the current implementation and arguments supporting answers.com.

The process, furthermore, also suffers from problems common to unstructured requirements elaboration, some of which can be seen in the extract:

- **Ambiguities** – It is not clear, for example, what AMO means in posts #2 and #3
o **Barrier to entry** – Any newcomer to the conversation, especially at later stages, will have difficulties sifting through unstructured textual posts to understand the current state of arguments and elaborations.

o **Over-specification** – It is a common mistake to over specify in requirements conversations. Post #16 is an example of this, where Basil suggests a possible implementation of Wikipedia searches and a means for designing it.

o **Unintelligibility** – Posts are not edited to improve intelligibility. An example of where this might be a problem is post #7, which is difficult to interpret due to poor sentence structure.

In Appendix C we have reconstructed on our framework the discussion in the extract, and the full elaboration process that followed. It demonstrates resolutions to the following problems:

- The goal graph represents the requirements in a *structured* and *concise* form, thus solving problems relating to a *lack of structure*, *noise* and *scattered content*.
- **Repetition is eliminated** since the same concept can only be represented once in the framework. Further, instead of being structured by time, *arguments are structured according to their relationship with the requirements*. These two factors solve the *soapbox problem*, since arguments that are repeated or heard last are not given precedence over their counterparts.
- Relating arguments to elaboration options *improves argumentation* since it discourages arguments from being lead astray onto other topics. Furthermore, arguments relating to different elaboration options are expressed separately, improving upon their clarity.
- Although not shown in the example in Appendix C, *ambiguities* can be dealt with by annotating them with an Issue as described earlier in this section.
- The **barrier to entry** is lowered since the goal graph, associated Issues and Arguments allow newcomers to clearly visualise the current state of requirements elaborations and discussions.
- A *goal-graph naturally discourages over-specification*, meaning that the requirements space can be more fully explored before tackling implementation issues.
- The clear visual representation of requirements and elaborations *improves intelligibility*. Furthermore, unintelligible additions to the model can be dealt with using the Issue annotations previously described in this section.

The following benefits should also be noted:

- The length of the *discussion was shortened* from over 200 posts to around 10 submissions, without loss of the key information relating to requirements elaboration.
The elaboration process can be replayed by stepping through each change to the model. This lowers the barrier to entry further, and provides a rationale resource for understanding why particular elaboration decisions were made.

These benefits, however, come with the price of an increase in the rigidity of the requirements elaboration process. An increase in a processes’ rigidity is commonly detrimental to its success (Fuggetta 2000), and therefore must be significantly outweighed by the benefits that the contribution provides. Open Source communities, for example, typically adhere to a highly flexible software development practices, and therefore the benefits of our framework will have to be substantial in order to encourage its adoption. This trade-off between structure and flexibility will continue to be a focal point as the contribution is further developed and validated.
5 Plan for Completion

This section outlines a plan for completion of the thesis. An overview of the remaining activities is first given, followed by details of immediate concerns, future enhancements, and validation of the contribution.

5.8 Overview

The remaining activities have been split into 5 groups, each of which has been allocated an approximate timescale. Diagram 5.1 shows a rough timeline for these activities, which are as follows:

Thesis write-up: This activity will take place iteratively, and has already begun. Chapters from this VIVA and other documents will be re-worked and included in the final thesis. Time will be left towards the end of the schedule to complete this task, and includes over-run time for the other activities. Appendix D contains a preliminary table of contents for the thesis, and gives the details of chapters that have already been worked on. This activity is expected to last the duration of the project, and is due for completion in September 2010.

Solidifying Current Framework: The framework presented in the previous section still needs to be made concrete. The sub-section below entitled Immediate Concerns covers the work that remains in this activity. This activity is expected to last approximately 3 months.

Enhancement of Framework: Possibilities exist for the provision of further benefits with the contribution, which may be necessary in order to justify its usage. The sub-section below entitled Future Enhancements covers the work that remains in this activity. This activity is expected to last approximately 4 months.

Development of Prototype: A prototype tool will be developed to validate the contribution. The sub-section below entitled Validation covers the development of the prototype, and the two remaining validation activities. This activity is expected to last approximately 8 months.

Validation on Small Case Studies: Small case studies, such as the example in Appendix C, have already been used to validate the framework during its conception. This iterative, test-driven development style will continue to be used, allowing for framework and prototype adjustments to be made in accordance with the feedback from these case studies. This activity will take place iteratively during the development of the framework and the prototype tool, and complements these activities.
**Validation on a Large Scale Case Study:** A large-scale case study will be used to demonstrate scalability and the contribution's benefits for an entire project. *This activity is expected to last approximately 5 months, allowing for the fact that the framework and prototype may need to be re-designed following the results of the case study.*

![Diagram 5.1 – Gantt chart: Thesis competition](image)

### 5.9 Immediate Concerns

The framework described in Section 4 is still work in progress, and its solidification is the next step in the thesis. The following issues are immediate concerns to be dealt with:

- Case studies have not fully explored the implications of multiple issues, especially where actions proposed for one issue affect another.
- The events that take place when selecting an Offline Issue have not been fully worked out, such as exactly where the information regarding the results of the offline activity is held.
- The precise semantics of Ignore Issue will also need to be explored, since at the moment it is not clear whether we will allow multiple instances, or just one with multiple pros and cons attached to it.
- The wording of terms in the framework may need to be re-thought. Offline Activity, for example, refers to an activity that is not directly concerned with the requirements model. Such an activity, however, could take place 'online' and hence using the term Offline Activity may cause confusion.
- We will look into extending the set of Issues to include semantic problems, such a contributor creating a goal that should have been a domain assumption.
The method used to date for framework development will continue to be used to explore and deal with these issues. This consists of the reconstruction of small case studies on the framework, which are analysed to identify alternative framework designs that provide more benefit. Having solidified the framework, it may be the case that further additions to the framework are needed in order to justify its adoption, which are discussed in the next subsection.

5.10 Future Enhancements

Further benefits may be required of the contribution in order to justify its adoption. A period of 4-5 months has been set-aside in the schedule to explore and develop these enhancements, some of which are already under consideration:

*Extended argumentation:* Further extension of asynchronous argumentation may yield more productive arguments. For example, discussants might be able to contest arguments made by others in a fashion similar to that of (Finkelstein 1992)

*Automatic issue placement:* Means exist for automatically placing Issues on a goal model. For example, words known to be ambiguous such as ‘security’ could be flagged with an ambiguous issue. Requirements inconsistency detection (Robinson et al. 2003) could be also used to place issues denoting logical inconsistencies.

*Change history analysis:* Automated analysis of the history of changes made to a requirements model could further benefit developers. It may be useful, for example, to analyse all changes associated with particular model element, thus providing rationale for why it was added to the model and the subsequent edits made to it.

*Associated risks:* Prioritising issues could prove to be beneficial to developers working on a requirements model with many issues. One technique for doing this would be to associate each issue with a risk value.

*Enhanced issue handling advice:* More advice could be given to support the proposal of actions to resolve issues. For example, specific types of offline activities could be recommended. We are also considering adaptive advice, whereby solutions that have previously worked for an issue within a similar context could be recommended.

Small-scale case studies will be used to explore, choose among, and develop these enhancements to the framework using the same techniques described in the previous subsection. The key issue in weighing up the advantages of alternative options will be that of benefits vs. rigidity, as discussed at the end of Section 4.
5.11 Validation

We have already begun validation of the framework by reconstructing small-scale case studies upon it. This technique will be used iteratively throughout the contribution’s development to provide constant feedback to the process. Likewise, small case studies will be used to develop the prototype tool in a test-driven style, by re-running them iteratively as the tool is constructed.

Small-scale case studies alone, however, will not justify the contribution. It is for this purpose that the prototype will be developed and used to reconstruct a larger scale case study, such the collaborative elaboration of requirements for a transport system (https://ovchip.cs.ru.nl/Main_Page). This will enable us to conduct a critical qualitative assessment of the framework’s viability for use in a large-scale project.

The success criteria for validation is the reduction of problems found in current asynchronous requirements elaboration techniques. These have been discussed in the problem statement and are summarised as follows:

- Hidden assumptions
- Ambiguities
- Requirements incompleteness
- Non-minimality of requirements
- Difficulties in change management
- Lack of stakeholder participation
- Barrier to enter
- Infeasible requirements
- Non-testable requirements
- Over-Specification
- Unintelligibility
- Requirements inconsistencies and conflicts
- Lack of structure
- The Soapbox problem
- Flawed argumentation
- Noise
- Scattered content

The benefit provided to developers elaborating requirements will be determined by the degree to which these problems are overcome. Further, we will argue that these benefits significantly outweigh the disadvantages that come with the adoption of a new technique, and a more structured process (Fuggetta 2000).

We will also compare our method against other works in the field that have been discussed in Section 2, such as the WikiWinWin tool (Yang et al. 2008). The case studies used to validate our contribution will be reconstructed on these tools and measured against the same criteria. We will then argue that the use of our framework solves significantly more of the problems identified than that of other techniques.
6 Appendix A – Envisioned Prototype

This section describes an envisioned prototype that utilises the conceptual framework proposed in this report. Each subsection chronologically follows events in the collaborative elaboration of a goal model for a meeting scheduler, adapted from an example in (A. Van Lamsweerde et al. 1995). The visual representation shown here is not part of the conceptual framework, but instead provides the reader with an example of how the framework might be implemented in a tool. Such a tool will later be developed as a proof of concept and for validation purposes.

6.1 Developers create a goal graph

Developers initially work together to create a goal graph composed of assertions and AND-refinements. Figure 6.1 shows the resulting visualisation in the prototypes interface, in which the cursor is over the Participants Constraints Known goal causing its in-node text to be displayed.

The means of editing the model and selecting various options is not described. We do, however, envisage that the tool would have an interface similar to that of DOORS or Objectiver, where a simple menu is placed on the left of the screen from which options can be chosen.

![Figure 6.1 – Prototype: Developers create a goal graph](image-url)
6.2 Annotating an assertion with an issue

A developer, Alice, believes that the goal Participants Constraints Known is over-optimistic, as there may be cases where participants are on holiday and cannot communicate their constraints. She therefore selects the goal and chooses to annotate it with an issue. She is then given a list of issues that may be placed upon a goal. From this she selects the Infeasible Issue type and enters a description of her reasons for annotating the goal.

The visual interface in Figure 6.2 shows the outcome of these actions, where the cursor is focused on the newly added Issue Annotation making its description appear. This shows that it is in the ‘in discussion’ state as it has been newly created.

Figure 6.2 – Prototype: Annotating an assertion with an issue
6.3 **Suggesting responses to an issue**

Developers may now select an issue and suggest responses. Figure 6.3 shows some suggested responses to the issue, in which the cursor is focused on an Offline Activity response making its description appear.

![Diagram showing suggested responses to an issue](image)

**Figure 6.3 – Prototype: Suggesting responses to an issue**
6.4 Suggesting a model alteration response

Bob believes that an alteration to the goal model would be the best response to the infeasible issue and wishes to suggest it. He therefore selects the issue and chooses the model transform response. He then enters *model transform mode*, in which he can suggest alterations to the model.

In model transform mode, he is allowed to perform edits on the goal model in much the same way as normal mode. Guidance is given, however, as to which types of edit might resolve the infeasible issue.

Once Bob is happy with the proposed model edits he exits model transform mode. The edits he has suggested are not applied to the model, but instead appear for others to view when the cursor is focused on the model transform response. This is shown in Figure 6.4, in which suggested additional model elements are in blue.

![Figure 6.4 – Prototype: Suggesting a model alteration response](image-url)
6.5 Arguing over issue responses

Developers may add arguments to issue responses by annotating them with pros and cons. This is shown in Figure 6.5 where the mouse cursor is focused on a suggested response, causing its associated arguments to be displayed.

Figure 6.5 – Prototype: Arguing over issue responses
6.6 Selecting an offline activity as a response

A developer with rights for selecting between responses to issues may change from normal mode into moderator mode. Charlie, who has these rights, feels satisfied that enough responses and arguments have been raised and decides to move forward by choosing an offline activity as a response to the infeasible issue. This moves the issue from the ‘in discussion’ state into the ‘pending offline activity’ state, and further actions relating to this issue are locked.

The visual display in the prototype at this stage is shown in Figure 6.6, in which the chosen offline activity response is highlighted in green. Collaborative elaborations of the goal model continue, but at this point we leave the example.

![Figure 6.6 – Prototype: Selecting an offline activity as a response](image)

Informal Def: We cannot guarantee that we will always be able to gather all participants’ constraints before a certain time. In some cases, for example, participants may be on holiday and therefore unable to comply.

Formal Def: N/A

State: Pending Offline Activity
Appendix B – Firefox Issue Ticket Extract

Below follows an extract from an Issue Ticket in the Firefox 3.0 development project that shows events typical of the requirements elaboration process. In this instance elaborations surround a requirement to add Wikipedia to the list of default search engines.

The extract contains the first 19 posts of a conversation that took 200 posts and over 2 years to complete. Ultimately, the decisions made in this issue ticket were reflected in the requirements document for Firefox 3.0. The full Issue ticket can be found at: https://bugzilla.mozilla.org/show_bug.cgi?id=380785

| A new issue ticket is created by Basil Hashem, entitled: “Add Wikipedia to the default search engine list” |
|------- Comment # 1 Rafael Ebron  2007-05-16 15:18:10 PDT -------|
| What’s the rationale? |
| For historical purposes, we didn’t put Wikipedia in at first because they didn’t have a search engine they were leveraging Yahoo and Google. |
| We added Answers.com in place of Dictionary.com and it also provided Thesaurus, Wikipedia, and other resources for answers. I don’t know what the usage is/was for Answers.com but still a pretty good service deal or no deal. |
| I suspect "Answers" doesn’t scream dictionary/thesaurus/wikipedia so usage might have been low? Don’t know. |
| Also note that "dict" was a keyword search for Answers.com in Firefox 1.5, not sure about 2.0. |

Rafael,

Thanks for the background.

The rationale is:
- Wikipedia has grown to become a relatively accurate, pretty valuable, mostly impartial, localized resource
- Wikipedia is a non-commercial entity that aligns well with Mozilla’s approach
- Wikipedia continues to grow and expand in multiple languages/locales
- Wikipedia’s knowledge base is larger (and growing) compared to answers.com
- Including both answers.com seems redundant

I think we should try to get some stats on answers.com searches if that’s possible and how often the wikipedia search plugin has been downloaded off AMO.

Comment #3 From Basil Hashem [:baz] 2007-07-09 11:37:49 PDT-------

AMO data shows that the Wikipedia search engine is the most downloaded search plugin from that site.

The following Firefox locales include answers.com:
af, ar, be, da, el, en-GB, en-ZA, es-AR, eu, ga-IE, hu, ka, ku, lt, mk, mn, nr, nso, sq, ss, st, sv-SE, tr, ts, ve, xh, zu

The following Firefox locales already include Wikipedia:
zh-TW, sl, sk, ru, ro, pt-PT, pt-BR, pl, nn-NO, nl, nb-NO, lt, it, fy-NL, fr, fi, eu, es-ES, de, ca, ar

1) We need to obtain permission from Wikipedia to use their search plugin and send traffic to their site (Mic Berman is reaching out)
2) Generate a list of locales that Wikipedia supports and map them to Firefox locales
3) For each locale, we should endeavor to use a search plugin that includes Search Suggest capability.

Comment #4 From Basil Hashem [:baz] 2007-07-09 11:39:46 PDT-------

Nom‘ing as blocking Firefox 3
Sorry, I meant wanted-firefox3, not blocking.

I use Wikipedia more often than I use a dictionary, but I still need a dictionary sometimes...

Firefox ships with "dict" keyword in en-US by default. If you type "dict pernicious" in the URL bar, you can quickly get definitions. Also, Google (assuming that that is your default engine includes a "define" keyword for quickly getting definitions.

We don't ship with any keyword search URLs in our bookmarks.html anymore for a bit now, at least since fx2.

I personally use the Answers.com and Wikipedia search engines side by side. Yes, Answers.com does include Wikipedia in their results, but I think it's a bad idea removing the dictionary/thesaurus abilities from the search bar.

I also think that having a search engine for a dictionary/thesaurus is infinitely more discoverable than a dict keyword. I'm willing to bet that the vast majority of the Firefox audience doesn't even know what keywords are, let alone that there may be some default ones included out of the box.

Czech (cs) Firefox doesn't ship neither with answers.com nor Wikipedia. However, I like the idea of replacing one current Czech search plugin with Wikipedia - mainly due to the fact 1 and 3 from comment #2.

I think that if you are good at googling, you already get a link to Wikipedia on the first page of Google search result. Also, adding Wikipedia to and removing Answers.com from search bar wouldn't be a better experience for users while Answers.com already include Wikipedia content.
(In reply to comment #2)

> The rationale is:
> - Wikipedia has grown to become a relatively accurate, pretty valuable, mostly impartial, localized resource
> - Wikipedia is a non-commercial entity that aligns well with Mozilla's approach
> - Wikipedia continues to grow and expand in multiple languages/locales
> - Wikipedia's knowledge base is larger (and growing) compared to answers.com
> - Including both answers.com seems redundant

I agree with all of these points, save for the last one. While including Wikipedia sounds like a great idea (especially for locales that have their own Wikipedias), I don't see why it has to come at the price of dropping Answers.com. Answers.com has, on top of Wikipedia, dozens, perhaps hundreds, of other sources - most notably general-purpose, as well as domain-specific, dictionaries.

Just one example:
Looking up "interest" on Answers.com (http://www.answers.com/interest) gives you, among others:
- A definition from the American Heritage Dictionary.
- Several definitions from Barron's financial dictionaries (which, I believe are not available elsewhere on the web).
- A legal definition from the Thomson Gale's Law Encyclopedia.
- Entries from the Columbia Encyclopedia and the Britannica Concise Encyclopedia.
- Translations to over a dozen languages.
- And, of course, the Wikipedia entry.

What this means for the user is more information, and more choice, which is something I think we are trying to promote.

[Full disclosure: I'm a former employee of Answers Corp., and as such I had a part in creating Answers.com. Although I no longer work at that company, I still enjoy their product as a user].
Comment #13 From Dwayne Bailey 2007-07-13 00:50:17 PDT

(In reply to comment #3)

> The following Firefox locales include answers.com:
> af, ar, be, da, el, en-GB, en-US, en-ZA, es-AR, eu, ga-IE, hu, ka,
> ku, lt, mk, mn, nr, nso, sq, ss, st, sv-SE, tn, tr, ts, ve, xh, zu

These locales: af, en-ZA, nr, nso, ss, st, tn, ts, ve, xh, zu are not attached to answers.com, we put those in to match en-US while we focus on translation of the UI. Users in South Africa would most probably prefer Wikipedia.

Comment #14 From Mark Tyndall 2007-07-24 08:40:44 PDT

(In reply to comment #3, and echoing comment #13)

> The following Firefox locales include answers.com:
> af, ar, be, da, el, en-GB, en-US, en-ZA, es-AR, eu, ga-IE, hu, ka,
> ku, lt, mk, mn, nr, nso, sq, ss, st, sv-SE, tn, tr, ts, ve, xh, zu

en-GB includes answers.com because Rafael requested it (bug 315385). I'd be exceedingly surprised if non-trivial numbers of UK users were using Answers.com.
> I agree with all of these points, save for the last one.
> While including Wikipedia sounds like a great idea (especially for locales that
> have their own Wikipedias), I don’t see why it has to come at the price of
> dropping Answers.com. Answers.com has, on top of Wikipedia, dozens, perhaps
> hundreds, of other sources - most notably general-purpose, as well as
> domain-specific, dictionaries.

I agree with Uri here and I think you should rename this bug to just adding Wikipedia to the search engine list. Removing Answers.com or another search engine from the default list should be a separate bug if necessary.

I don’t think you have enough information to arbitrarily drop Answers.com from the list like this, and they do provide much more than just "user-generated" encyclopedia info.

I recommend that we keep Answers.com until you have real data that says otherwise or they do something "evil". Have they done anything bad to piss off our users that we don’t know about? Barring that, Answers.com should stay.

Also, it should be noted that they’ve been good citizens with their promotion of Firefox and development of Firefox extensions. It’s not like they’re Yahoo! or anything (oops, did I say that out loud?).

This bug has been a bit stale and we need to make a decision. Here’s my recommendation.

0. I’ve retitled the bug to say "Add Wikipedia to default search engine list" - removed reference to answers.com.
1. Where Wikipedia does not appear in a Firefox locale, add it (assuming there’s good localized content and the localizers agree that it’s worthwhile).
2. If Wikipedia is there, ensure that we use the "Search Suggest" version of the search plugin, if it’s supported by Wikipedia for that language.
3. It should be a localizer decision as to where in the list Wikipedia appears. Localizers should be aware of various contractual constraints with regard to Yahoo and Google.

We won’t remove the answer.com plugin for now and will try to work with the company to get data about degree of Firefox usage (per locale).

Any questions?
<table>
<thead>
<tr>
<th>Comment #17 From Jesse Ruderman 2007-07-31 11:39:01 PDT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Does Wikipedia have a search mode where if an article (or redirect) exists, it goes there, but if it doesn't, it searches instead? I don’t find Wikipedia's search very useful, so I usually either (1) guess at the article name (en.wikipedia.org/wiki/foo) or (2) search Google for 'wikipedia foo'.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Comment #18 From Gavin Sharp 2007-07-31 11:54:14 PDT</th>
</tr>
</thead>
<tbody>
<tr>
<td>(In reply to comment #17)</td>
</tr>
<tr>
<td>&gt; Does Wikipedia have a search mode where if an article (or redirect) exists, it goes there, but if it doesn't, it searches instead?</td>
</tr>
<tr>
<td>That's essentially the difference between the &quot;Go&quot; and the &quot;Search&quot; buttons in it's search form, as far as I can tell, and as far as I know that search plugin uses the &quot;Go&quot; mode which behaves as you describe (<a href="http://en.wikipedia.org/w/index.php?title=Special:Search&amp;search=">http://en.wikipedia.org/w/index.php?title=Special:Search&amp;search=</a>&lt;input&gt;, for example; &quot;Cow&quot; sends me directly to <a href="http://en.wikipedia.org/wiki/Cow">http://en.wikipedia.org/wiki/Cow</a>, but &quot;fowoo&quot; sends me to <a href="http://en.wikipedia.org/wiki/Special:Search?search=fowoo">http://en.wikipedia.org/wiki/Special:Search?search=fowoo</a>).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Comment #19 From Basil Hashem [baz] 2007-07-31 12:30:18 PDT</th>
</tr>
</thead>
<tbody>
<tr>
<td>I can't tell what algorithm the search suggest version of Wikipedia is using, but it seems to do what I would expect and want. It seems to collect pure alphabetic results...but not quite.</td>
</tr>
<tr>
<td>E.g. I do a search for &quot;Milo&quot; in the search plugin, I get search suggestions that don't appear when I do a &quot;Go&quot; search for &quot;Milo&quot; on the website. The nice part is if you provide something disambiguated, e.g. &quot;Milos Forman&quot;, you'll get the exact page and don't need to worry if it was a Go vs. Search search.</td>
</tr>
</tbody>
</table>
The example in this appendix shows how a set of requirements elaboration events in the Firefox project might have taken place using our framework. An extract of the original set events that took place in the Firefox development Issue Tracking system is given in Appendix B.

David believes that Wikipedia should be included among the set of default searches in Firefox, and therefore adds a goal stating this to the relevant section of the goal graph.

Please note that the whole goal graph is not shown for the purpose of simplicity.
Blake believes that there are already enough default search options, and that the addition of the new goal results in a non-minimal reduction. He annotates the graph with an issue accordingly.

Furthermore, Blake believes that the new goal should be removed as the goal graph already specifies the inclusion of dictionary.com with the default search options, which contains both a dictionary and an encyclopaedia. He therefore proposes a model transform action as a resolution to the issue, and attaches his argument to it.
Ian does not believe that the suggested model transform is a good choice, as Wikipedia ranks 10th on the most visited web pages on the internet and it would therefore be of benefit to Firefox users to be able to search the site directly from the search bar.

Type: Model Transform
Pro: dictionary.com already has an encyclopaedia
Con: Wikipedia is one of the most visited web pages and therefore should be readily accessible in the search bar

Deliver the best possible browsing experience

Provide a good set of default search options

Include dictionary.com among the default search options

Include Wikipedia among the default search options

Remove Wikipedia Search Option

Too many default searches
Basil believes that the goal to include answers.com should be removed, as Wikipedia is more popular than answers.com and provides dictionary definitions of words. He proposes a model transform accordingly, to which he adds some further ‘pro’ arguments.

Type: Model Transform
Pro: Wikipedia already contains a dictionary, and is more popular than dictionary.com
Pro: Wikipedia’s knowledge base is much larger than that of dictionary.com
Pro: Wikipedia conforms to the open source ideology, while dictionary.com does not
Basil spots a possible inconsistency in the model introduced by the Wikipedia search requirement. Namely, the increased traffic to Wikipedia could be unwanted by the Wikipedia team. He suggests contacting the Wikipedia organisation to ask them if this will be a problem.

Basil, acting as a moderator, chooses to select the offline activity suggested in the previous event, therefore locking the issue. Mike follows up the offline activity by contacting the chief technology officer at Wikipedia, who replies stating that the Wikipedia organisation would be pleased to receive the extra traffic.
Mike then marks the offline activity as completed, thus re-opening the issue. He also proposes to ignore the issue given the reply he has received from the Wikipedia organisation.

Basil decides that given the results of the offline activity the remove issue action should be taken. He therefore acts as a moderator and selects the suggestion, thereby removing the issue from the model.
Basil considers the pros and cons of the proposed actions associated with the remaining issue. He then decides to move forward elaboration by choosing to remove the dictionary.com search option, thus resolving the issue.

Include Wikipedia among the default search options

Provide a good set of default search options

Avoid disputes with other organisations
Below follows the preliminary table contents for the thesis that has been adapted from (Nentwich 2004). The contents of chapter 2 will be taken from this document and the first year VIVA (Fitzgerald 2008). This document also contains elements that can be extracted to form parts of chapters 2 and 3. The remaining chapters will all need to be completed as the thesis progresses.

1. Introduction
   1.1. Overview of Contributions
   1.2. Statement of Originality
   1.3. Thesis Outline
2. Motivation
   2.1. Requirements Engineering in Context
   2.2. Goal-oriented RE
   2.3. Collaborative RE
   2.4. Asynchronous Communication
   2.5. Related Work
      2.5.1. Requirements Elaboration
      2.5.2. Collaborative Requirements Elaboration
      2.5.3. Requirements Inspection and Critique
      2.5.4. Sense-making Technologies
   2.6. Chapter Summary
3. Overview
   3.1. Approach
   3.2. Framework and Schema
   3.3. Lifecycle of a Requirements Model
   3.4. Introducing a Running Example
   3.5. Chapter Summary
4. Asynchronous Requirements Elaboration
   4.1. Critiquing a Goal Model
   4.2. Proposing Responses to a Critique
   4.3. Arguing Over Alternate Responses
   4.4. Selection and Application of Responses
   4.5. Chapter Summary
5. Implementation
   5.1. Overview of Prototype
   5.2. Environment
   5.3. Advanced Features
5.3.1. Graphical Display
5.3.2. Change History
5.3.3. User Permissions

5.4. Chapter Summary

6. Case Studies
6.1. Case Study: Meeting Scheduler
6.2. Case Study: The Firefox Project
   6.2.1. Experimental Set-up
   6.2.2. Results
   6.2.3. Evaluation
   6.2.4. Conclusions

7. Scalability
   7.1.1. Large Projects in Context
   7.1.2. Scaling Collaboration
   7.1.3. Scaling Model Complexity
   7.1.4. Chapter Summary

8. Related Work Revisited
9. Future Work
10. Conclusions
References


Shum, S.B. et al., 2006. From gIBIS to MEMETIC: Evolving a research vision into a practical tool. *Design Rationale Workshop: Design, Computing & Cognition Conference*. 
