An Exploration of how Representational Tools can Support Creativity and Relational Reasoning

By

ANDREAS KJØNG SJÖBLOM
NIKOLAJ MARIMO STØVRING

AALBORG UNIVERSITY
STUDENT REPORT

Master Thesis Medialogy
AALBORG UNIVERSITY, COPENHAGEN
INSTITUTE FOR ARCHITECTURE AND MEDIA TECHNOLOGY

SUPERVISOR: OLGA TIMCENKO

MAY 2019
Abstract

In this thesis, we perform an exploratory investigation into relational reasoning, creativity, and how a representational tool can support both. Gathering and Mediating information is a process which requires skill in organizing and structuring information, and creativity and relational reasoning are practised when perceiving patterns in information. A representational tool is a technique or application which allows the user to map information into a graph-like network structure, for memorization and learning and collaboration. The application developed will enable users to create representational artefacts based on the ArcForm notation. The app is integrated with, which allows users to gather information. To promote relational reasoning and creativity, the application is integrated with ConceptNet. To visualize the information, the app is based on the ArcForm Notation. Usability tests were performed on the app to determine its viability.
Contents

1 Introduction .......................................................... 1

2 State of the Art ..................................................... 3
   2.1 Project Management Software ................................. 3
   2.2 Knowledge Mapping Software ......................... 3

3 Analysis ............................................................... 5
   3.1 Idea Representation ......................................... 5
      3.1.1 Diagrammatic Representation ............................ 6
      3.1.2 Representational guidance ............................. 9
      3.1.3 Creativity .............................................. 11
   3.2 Relational Reasoning ......................................... 12
   3.3 ArcForm ..................................................... 12
      3.3.1 Design Requirements .................................... 15

4 Methodology .......................................................... 17
   4.1 Research Strategy ............................................ 17
   4.2 Target Group ................................................ 17
   4.3 Data Collection and Data Analysis ......................... 18
   4.4 Experiment .................................................. 18

5 Iteration 1 ............................................................ 21
   5.1 Introduction .................................................. 21
      5.1.1 The exploratory iteration ............................ 21
   5.2 Design ......................................................... 22
      5.2.1 User Interaction ........................................ 22
      5.2.2 ArcForm representation ................................ 22
   5.3 Implementation ............................................... 23
      5.3.1 Map Creation .......................................... 24
      5.3.2 Connecting Tokens with Arcs ....................... 25
      5.3.3 Labelling .............................................. 27
   5.4 Evaluation .................................................... 28
6 Iteration 2

6.1 Introduction .................................................. 31
6.2 Re-Design ......................................................... 31
6.3 Implementation .................................................. 32
  6.3.1 ConceptNet & DBPedia integration ......................... 32
  6.3.2 Dynamic Menu ............................................. 33
  6.3.3 Force Directed Graph ..................................... 34
  6.3.4 Test Manager ............................................... 35
6.4 Evaluation ....................................................... 36
  6.4.1 Test Setup ................................................ 36
  6.4.2 Test Procedure ........................................... 37
6.5 Results ......................................................... 37
6.6 Conclusion ..................................................... 38

7 Iteration 3

7.1 Introduction ..................................................... 39
7.2 Re-Design ...................................................... 39
7.3 Implementation .................................................. 40
  7.3.1 Grid based Layout ....................................... 40
  7.3.2 Color Coding .............................................. 41
7.4 Evaluation ....................................................... 42
  7.4.1 Test steps ............................................... 42
7.5 Results ........................................................ 44
7.6 Conclusion ..................................................... 45

8 Discussion

8.1 Project direction ............................................... 47
8.2 Evaluation ...................................................... 48

9 Conclusion .................................................... 49

10 Future Works .................................................. 51

Bibliography ...................................................... 53

A 2nd & 3rd Iteration Scenarios .................................. 55
Contents

B 2nd & 3rd Iteration Evaluation Questions

C 2nd & 3rd Iteration Participant Responses

C.1 2nd Iteration interview responses
   C.1.1 Participant 1
   C.1.2 Participant 2
   C.1.3 Participant 3
   C.1.4 Participant 4
   C.1.5 Participant 5

C.2 3rd Iteration interview responses
   C.2.1 Participant 1
   C.2.2 Participant 2
   C.2.3 Participant 3
   C.2.4 Participant 4
   C.2.5 Participant 5
   C.2.6 Participant 6
   C.2.7 Participant 7
Chapter 1

Introduction

Information gathering is a fundamentally important ability for researchers. The ability to search, gather and synthesize information, to then mediate to fellow researchers is a focal point of research work-flow, and extensive research, in research methods and evaluations are performed every day. Yet information technology which directly assists in the information gathering phase is strangely absent.

A common description of an undergraduates research flow involves using Google to search for relevant terms in scholar and within the references of references, and the accumulated information is stored within a research document meant to bring structure, and overview to the information.

In this phase project management software or shared folders are used to keep track of resources, and storing the relevant information with related and associated topics can be an arduous process.

Organizing and structuring information gathered, in such a way that it’s easy to use and browse, requires consistent and structured organization methods.

An example case is found within the research phase where multiple articles from across multiple topics must be collected, read and mediated through, resumes, articles and videos.

The current way of achieving this requires researchers to be able to browse university library portals, various search engines and wikis. Tools such as ReadCube, Citavi and Zotero exist to assist in this endeavour of maintaining overview. Yet the multitudes of information, relevant and irrelevant, is processed and evaluated, through writing it into a document.

This can quickly become complicated when a team of researchers work together to investigate various aspects of a topic, and need to be able to mediate and synthesize their findings.

1https://app.readcube.com/
2https://www.citavi.com/en
3https://www.zotero.org/
Chapter 1. Introduction

The problem we observed can be boiled down to two aspects of research flow. One, Systematic gathering of related information in relation to a problem statement can quickly lead to a loss of overview. Two mediation of gathered knowledge requires extensive triangulation, source verification and knowledge mapping.

Research in the field of knowledge mapping and sharing has been investigated for many years. In 1990 Novak[11] introduced the concept of concept maps, -a precise tool for describing the relationships between concepts. The positive effects of using Concept for learning, memorization and knowledge sharing have been researched extensively [13][8][10], yet the fundamental application of using diagrams to gather information seems unexplored.

In this paper we introduce a line of research into how representational tools can support idea generation in the context of creating research papers. As our motivation is to create a representational tool meant to support idea generation, choosing an adequate representational notation is important to base it on.
Chapter 2

State of the Art

In these sections, we will investigate the various applications developed to assist people with project management and digital tools available to visualize knowledge through mind maps, databases, and graphs.

2.1 Project Management Software

Project management software such as Microsoft Projects\(^1\) Jira\(^2\) Trello\(^3\) or Microsoft Teams\(^4\) function as interfaces to help keep track of project deadlines and tasks, and function as hubs of resources relating to a specific projects. These implementation tools are incredibly useful in project management yet as the focus of the application is on project management and not information management, the interfaces involved in aggregating information rely on the user collecting and structuring the information.

2.2 Knowledge Mapping Software

Knowledge maps are visual techniques developed to memorize, understand and mediate complex topics through the use of diagrams. From a usability perspective, current knowledge mapping tools provide no innovation in the workflow behind visualizing knowledge. It is puzzling, that while structuring and organizing information has moved from paper to digital documents, the workflow behind researching and writing papers yet remains tied to the process of manually gathering and structuring knowledge within papers. Especially when considering the advent of machine learning techniques and relational databases.

\(^1\)https://www.microsoft.com/da-dk/store/collections/project?cat0=devices
\(^2\)https://tinyurl.com/y2sc2l53
\(^3\)https://trello.com/en
\(^4\)https://tinyurl.com/MicroTeams
Interactive applications which allow users to visualize knowledge through mind maps are commonly distributed in various software suites. Mind mapping software such as Xmind\textsuperscript{2.2} and LucidChart\textsuperscript{2.1} provide the user with the functionality to create mind maps collaboratively, and this basic functionality works as intended, yet none of these applications explicitly give access to information gathering services.

Structuring, and gathering information exists as two separate processes which currently require two different types of applications. Which means that application which enables the user to search, organise and mediate knowledge does not seem to exist in one unified form.
Chapter 3

Analysis

Developing and sharing ideas are fundamental aspects of generating new knowledge, and in the evolution of the domain of information and knowledge mapping, many complementary visualisation techniques have been developed.

When researching any topic, the process often involves aggregating and structuring information. This consists of browsing the web for related topics, recording notes and bookmarks and writing commonly structured sequentially within a paper.

A common way to mitigate the learning process is to use conceptual diagrams to explain complex processes. Conceptual diagrams take advantage of our cognitive visuospatial ability\[2\], to describe the relationships between concepts.

The nature of conceptual diagrams tends to give users an overview of the collection of information in such a way that has proven to improve the learning outcomes during collaborative learning and knowledge sharing.

In the following chapter, we will explore how conceptual diagrams can be analysed and created, we will discuss how they relate to knowledge overview, creativity and relational reasoning, and how an application based on a novel concept diagram notation could support all three.

3.1 Idea Representation

The type of information visualisation tool to be used is largely a function of the purpose for which it is intended.

In general, the purpose of mapping techniques is to allow users to represent or manipulate complex relations within diagrams.

The representations allow users to share and discuss the components and the relations between, and to gain deep level understandings of concepts. Furthermore, compared to verbal and written descriptions, the contiguity of components add additional layers of meanings to the idea \[13\].
Furthermore, studies have shown that visual displays enhance learning through the dual nature of the encoding required to create the visualisations. That is to say, the multi-modality involved by requiring the user to determine where to place components as well as considering why the relationships support the placement assists in the retention of the information. [8]

3.1.1 Diagrammatic Representation

Diagrammatic visualisation techniques such as mind maps and concept maps have for many years been acknowledged as a useful tool for improving learning outcomes for students. The general theory supporting this is that students can analyse and represent complex ideas through diagrams several positive factors such as: enhancing, retaining and improving knowledge. In collaborative contexts, the phenomenon called joint retention, which describes the enhanced learning experienced when using multiple modalities in the learning experience, has also been observed. The case of information mapping occurs when visual and verbal representations of knowledge are retained synchronously[7]. Furthermore, the active engagement required of the person creating the representation also provides greater learning. [7].

Mind Maps

Mind maps are also known as association mapping, or idea mapping is a non-linear representation of ideas and associations [1]. The advantages of mind mapping lie in the unrestrained nature of the associations which can be drawn. Any idea can be connected to any, and improvisational and spontaneous ideas are encouraged. Creative ideas and associations can quickly be visualised, which makes mind-mapping especially useful for brainstorming sessions where all ideas should be accepted. They also serve well as mnemonic devices as remembering a diagram tends to be easier than memorising a text [7]. The disadvantages of mind mapping are, on the other hand, a consequence of this unrestrained feature, as the mind-maps are often difficult to understand by others than the creator. Since concepts are connected by association, mind mapping is also limited when attempting to explain complex topics. In such cases relational analysis is more useful [7].

In conclusion, mind maps are handy for brainstorming, but their limitations are clear when attempting to share or represent complex ideas.

Concept Maps

Concept maps or Novak Maps are representations of ideas with a focus in the relationship between concepts. Often there is a hierarchy in place and how one object influences another in the hierarchy is clear. This means that concept maps tend
3.1. Idea Representation

Figure 3.1: Mind Map [https://tinyurl.com/y5mx994o](https://tinyurl.com/y5mx994o)

...to describe the inner workings of any system with higher precision than mind-maps[7]. The advantages therefore are, those very complex concepts can be explained in detail with the use of concept maps. In a learning context concept maps are very useful, as meaningful learning can take place when linking new information to existing knowledge [6][10].

Image link

The disadvantages to concept maps are that they due to their complexity, may be more challenging to memorise, and they can be intimidating for viewers. Furthermore, the rigid workflow in which concept maps are constructed makes them more challenging to learn how to create. Finally, the linear hierarchical system can be inadequate when describing cyclical concepts.

In conclusion, concept maps are useful for describing complex concepts with utmost precision, but due to the nature of identifying how concepts relate to one another, they are more challenging to construct.

Argument Maps

Argument maps are a relatively modern form of knowledge mapping which focuses on the argumentative aspect of idea collaboration and sharing. These maps represent concepts as claims or objections, which consequently can be supported or contended by additional claims and objections. Argument maps are extremely useful in determining the cause and effect relationships behind events and actions taken as every claim and objection must be recursively substantiated. Finally, at the top level are "basis boxes" which contain evidence of some sort which can help support or disprove claims[7].

The disadvantages of argument maps are, that when creating one, it is assumed
Chapter 3. Analysis

Figure 3.2: Concept Map
https://www.cooper.com/journal/2016/8/
concept-mapping-for-designers-of-the-future

Figure 3.3: Argument Map
https://tinyurl.com/y6agaxev
that all of the information required to support and disprove claims is accessible to the creator. Furthermore, unless the user understands the levels of abstractions required to create objections or claims, the argument map may remain quite simple in its conclusions. That is to say, if the argument position, for example, is "Women are better off than men" the argument map may end quickly if the creators have a limited understanding of the socio-economic variables which may influence the answer[7].

In conclusion, the various is focused on the idea of types of techniques available have several pros and cons and as discussed by Eppler et al.[8], as the different techniques are more useful depending on the context in which they are employed.

The field of information mapping contains even further a large number of applications and techniques meant to improve the effectiveness of idea generation and communication, through diagrams, flowcharts, mind maps etc. As we intend to develop our type of representational tool, we will, in the following section, from a theoretical standpoint, examine how the design of a visualisation tool influences the specific learning outcome.

3.1.2 Representational guidance

Suthers[13] describes the creation of representational tools as part of process called *representational guidance* which is described in figure 3.4.

Representational guidance is the influence which the representational notation and the representational tool has upon the representational artefact. In other words, what the notation and the tool consider salient, guide what will be valuable in the representational artefacts the user creates. The types of representational notation can be domain specific or general[13]. For example, mind-maps and brainstorm diagrams can be created across all domains, and on the other hand, UML
and class diagrams are explicitly created in relation to Computer Science. The differ-
et types of notations and tools required to understand to create either, require
the user to think about the way they categorise their knowledge.

The representational notation is a set of rules and primitives which describe
how they interact and what types of relationships can exist between them.

A Representational tool mediates collaborative learning interactions by providing learn-
ers with the means to express their knowledge in a persistent medium. representational
Guidance constraints which knowledge can be expressed and therefore emphasises specific
topics

Representational tools cover a broad spectrum of types and use from inform-
ation mapping software, simulation and data visualisation. The spectrum of
representational tools also covers different graphs structures such as mind maps,
hierarchies, family trees etc.

The purpose of representational tools is to serve as cognitive abstractions, which
allow users to see patterns, discover abstractions and observe relationships be-
tween objects [13].

The representational artefact is the output of the representational tool; it is the
actual diagram/graph which the users create, describe and share.

Therefore, to understand a representational artefact, one must understand the
notation, and to create the artefact, one must understand how to use the tool.

Information visualisation techniques can be analysed using Suthers model of
representational guidance to determine which constraints and salience different
visualisation/mapping tools have. According to [13] the different way in which
the tools present where and when information should be inserted, profoundly
influences how a user interacts with the tool and how the corresponding artefact
looks.

In conclusion, when considering which representational notation use, and how
we will develop the tool, we must consider how the representational guidance will
influence how the user interacts with the tool, as well as how the artefact will be
created.

In this paper, we introduce a line of research into how representational tools
can support idea generation in the context of creating research papers. As our
motivation is to create a representational tool, meant to support idea generation,
choosing an adequate representational notation to base it on is essential.

We believe that to share knowledge, it is imperative that the notation used is
easy to read and understand, yet also flexible enough to allow complex ideas to be
represented without loss of overview.

We will now explore which qualities we would like to consider in regards to
the representational guidance of the tool. First, we will investigate how creativity
can be encouraged in collaborative idea generation activities. Secondly, we will
explore how relational reasoning can be used to help solve problems, and how it
could be possible to support it within the application.

### 3.1.3 Creativity

Creative ideas have always been a driving force behind innovation and studies, within the development of creative ideas from a psychological standpoint, are numerous.

The novelty of an idea and its usefulness often diverge [3]. Based on the experiments described in The primal mark: How the beginning shapes the end in the development of creative ideas [3], whenever an idea is seeded by a concept, that concept works as a “primal mark” which influences the trajectory of the idea. In the paper, they determine that primal marks, which people are familiar with, create useful yet less novel ideas while primal marks, which the users are unfamiliar with, create novel yet useless ideas. Based on their research, integrative primal marks, which foster both familiar and unfamiliar content, create the optimal conditions for ideas with high novelty and high usefulness [3].

The research and creativity phase can furthermore be categorised into four distinct phases. The four phases are generation, elaboration, championing and implementation, which are facilitated by network connections [12].

Perry-Smith [12], describe the four phases of idea generation which cover the factors involved when an idea is conceived until an implementation can be brought forth [12].

The four phases are idea generation, elaboration, championing and implementation.

In this paper, we will focus on the idea generation phase. This phase is described as a process in which variations in associations are combined and recombined and the creator self-selects a single idea [12]. This is different from brainstorming as the purpose is not to develop a large number of ideas, regardless of novelty and usefulness, and it concludes when the user deems the idea somehow novel, useful and promising. This idea is often vague and is elaborated upon in future phases. This process is often unconscious and unpredictable, yet related information and environmental stimuli are expected to facilitate the process [5]. On the other hand, a large amount of knowledge in a field may, on the other hand, constraint, which associations are taken under consideration [12].

Based on this information, we believe that implementing a method within the application to present the user with related and associated concepts to the subject they are working on function as a catalyst for creativity.

Providing users with associated relationships may not be enough to facilitate idea generation, as the ability to perceive patterns within seemingly unrelated information must be supported.
3.2 Relational Reasoning

This ability is described as relational reasoning and is as a logical instrument very useful in problem-solving, critical thinking and analytic reasoning. According to Alexander et al.[1], relational reasoning is fundamentally the capability to detect patterns in within information. This allows us to discern conflicting evidence and draw conclusions across domains and is a foundation of learning[1]. Relational reasoning covers four categories of analytic reasoning, analogy, anomaly, antimony and antithesis.

Analogous reasoning covers our ability to understand, construct and establish relationships between concepts through metaphors and analogies. This is the fundamental perception which allows us to recognise similarities between concepts[1].

Anomalous reasoning involves discerning a pattern among sets of objects, ideas or situations, as in Analogous reasoning. It is fundamentally present when we detect a break from that pattern. Anomalous reasoning is used to detect inconsistencies in thinking[1].

Antimopy is the ability to detect illogical inconsistencies; for example, the statement "There is no absolute truth" is in of itself illogical as it claims to be an absolute truth about truths.

Antithetical reasoning is the ability to recognise contrasting viewpoints and synthesise new opinions which reveal contradictions[1]. For example "All mammals give birth" and "The platypus lays eggs" by examining the fact that a Platypus is defined as a mammal a logical inconsistency is detected.

According to Alexander et al., relational reasoning is pervasive and used continuously and applied by people of all ages and demographics, and it is a skill which can be taught[1].

In conclusion, relational reasoning is the ability relational reasoning could be possible, by encouraging the user to detect anomalies across patterns, construct metaphors and discern logical inconsistencies, and we believe that the representational guidance of the tool can be designed to support this.

In the following section, we will describe the representational notation ArcForm and how it could be used as a basis for an application which supports creativity and relational reasoning.

3.3 ArcForm

ArcForm is a notation which means to represent concepts in their most abstract to concrete form it was developed by Allsopp[2] as representational notation meant to allow users to represent thoughts and ideas through a graph like a network. This is accomplished by the employment of two distinctions. And the object of thought (object) and a thought[2].
The notation is built around this system of thoughts and object and on the surface, it looks like a regular graph structure which uses objects and arrows to represent connections.

The ArcForm notation consists of two types of primitives, the thought and the object. The Arcs are arrows which point from token to token to describe what sort of relationship the subjects of the tokens have.

An object can be anything which can describe as the subject of a sentence; therefore, it is possible for an object to be a physical object, an event, an idea etc.

A thought is a declarative statement about an object; for instance, Paris is an object, "Paris is a City" is a thought which involves the objects Paris and City.

**First Order Thoughts**

"Paris is a City" is referred to as a first-order thought, one variation of a first order thought is thought such as "Jane sings". In this case, the Arc does not connect to objects as "sings" is a descriptive attribute of the object "Jane".

**Higher Order Thoughts**

Higher-order thoughts are thoughts in which multiple first order thoughts are combined to create more nuanced statements. For instance "Jane Sings Beautifully", in this case, "Beautifully" describes how the thought "Sings" is performed by the object "Jane".

The composition of higher order thoughts can continue indefinitely to describe anything.

An additional thought variation is the collection object, which functions as the "and" statement, for example, "Jane" and "Doug" and "John" are friends with "Jim", and so on.
can be represented with fewer Arcs, by using a collection token.

In summary, the various features of the primitives are as follows:

- Thoughts can point from one object to another object for declarative statements
- Thoughts can point to an object for descriptive attributes
- Thoughts can be nested for higher-order thoughts
- Objects can be nested in Collection Objects

In the rest of this paper, we will refer to the objects as tokens and thoughts as Arcs.

An additional feature is that Arcs can point and originate from other Arcs, which allows statements to be concatenated. This means that the granularity of the map can be dynamic depending on the needs of the user. (dynamic complexity)

To understand why ArcForm is interesting to this project, we must look into the notations three distinguishing features.

**Natural Language Interface**

Since Arcs represent meaning between two tokens, the structure is similar to how the natural English language is processed. Therefore learning how to read and process the meaning of an ArcForm map requires little training.

This lowers the bar of entry for any user of the notation and allow

**Unitokenality**

As every ArcForm statement has no requirement for orientation or layout, any token can be represented a single time, this means that every concept can be uniquely represented. In a geographical map Paris, Berlin and Copenhagen are only represented once, and the location of each city can be found in relation to the other, in the same manner the tokens in an ArcForm map can contain single instances of an abstract concept, located relevantly spatially.

**Structured Data**

Because ArcForm statements can be completely generic, means that any implementation of ArcForm needs to translate into software the functionality of the Arcs, tokens, and the collection token to create a foundation for any representational tool based on ArcForm.

We decided to use the ArcForm notation as the basis for the application since the structure of the notation is almost built to be digitally adapted, in contrast to the traditional methods.
How do we create a representational tool based on ArcForm, which supports relational reasoning and creativity by providing external information with an intuitive user interface?

3.3.1 Design Requirements

The application must be a representational tool based on the ArcForm notation. The representational tool must support relational reasoning by allowing the user to determine relationships through antimony, anomaly and antithesis. The representational tool must support creativity with external information.
Chapter 4

Methodology

This section will state choices in research strategy, methods, techniques and procedures, along with a final list of requirements for answering the problem statement.

4.1 Research Strategy

The project is venturing in an exploratory direction as this research tries to develop an application which supports relational reasoning, creativity and overview with the use of a representational tool. The grounded theory provided in the analysis section 3, the design requirements 3.3.1 and the framing of the project, suggests an inductive approach with the use of mixed methods [bj\IeC {\o }rner2015]. To answer our hypothesis, the application that supports the theory has to undertake several revisions before being deemed stable enough to conduct an evaluation. Once a stable version of the application with all its design requirements checked off as ready, evaluation can begin.

The degree at which the prototype representation is acceptable can be incredibly subjective and is to some extent determined by a participant’s technical expertise. To evaluate the prototype and understand the user intent, we will use a combination of qualitative interviews and usability testing. The application will record mouse behaviour and task times, meaning how long it takes a user to complete an array of specified tasks. Observing a participant performing tasks, and how they interact with the application, can help determine if they use the application as it is intended.

4.2 Target Group

The intended use of the application is, to help a user organise and maintain projects 3.3.1. To establish overview, organise research directions, citations, and in general be source critical; is a few topics which researchers need to control. To
form and improve the application in this direction, the target group for this project is researchers. Participants for the evaluation are convenience sampled at Aalborg University, Copenhagen.

4.3 Data Collection and Data Analysis

A usability test which guides the participant through a series of nine tasks created to test all functions and aspects of the application. The idea is to observe how people use the application realistically, to identify problem areas, and what people enjoy. The list of tasks is found in the appendix. Three different scenarios, randomly ordered, ensures any novelty bias from experiencing the scenarios in the same order is avoided, and it considers the application learning curve. While the test participant performs the associated tasks to the given scenario, a research observer can dynamically verify the application. The application has scripted instructions which the participant receive as pop up messages, and is automatically logging the time it takes to accomplish a task. The task information window is moved to the top right corner, so a test participant at any given time can read the instruction. It will ensure minimal test distraction by the test conductor. To review test interaction, a screen recording application will be used.

An informal interview is held between each of the three scenarios. The participant is asked a set of questions in regards to their current opinion of the application and its general usability. After the first scenario and throughout the rest of the test, the participants are asked if their views have changed. In the end of the questioning after the third scenario, participants are asked what the program is used for.

The qualitative interviews supported with the quantitative usability tests met-rics gathered are expected to inform the researchers on reaction times, amount of actions used to complete the task, ratios of success to failure to complete the task, along with user motivations, number of times user seems frustrated etc.

The first iteration test in this project will focus on fine-tuning intuitiveness and analyse user interaction, while the second iteration will focus on a performance test of the application and usability.

4.4 Experiment

A participant should go through all possible actions and types of interactions the system supports. Below is a list of actions available to the user.

1. Navigate the canvas or, including zoom in/out.
2. Search within a database.
3. Create nodes and relational arcs.
4. Delete nodes and relational arcs.
5. Multiple ways of doing the above, e.g. using mouse or keyboard shortcuts.
6. Create a representational artefact.

Design of the test experiment should include all of the above including a more ambiguous task, so a test participant has to combine all of the actions to complete a task, giving a more realistic use of the application.

The application went through a series of iterations based on the iteration cycle methodology. These iterations are described in the following chapters.
Chapter 5

Iteration 1

5.1 Introduction

This iteration focuses on building a prototype with established design requirements from the analysis in mind. Expected user interactions are based on the ArcForm notation and are formed in a list. The list can be seen in section (ref to user interaction list in appendix). This list became the main design and implementation guideline for creating a functioning prototype with basic user- and database interaction.

The goal of this iteration is to develop a working prototype which enables the user to create ArcMap. This means we will answer the following questions in the design section.

- How will the user create Tokens and Arcs?
- How will the application display Tokens and Arcs?

The approach we take to answer these questions are exploratory in nature and decisions in regards to which implementations will be based on responses from expert interviews.

5.1.1 The exploratory iteration

The ArcForm notation consists of many natural language rules and map generation constraints based on the English language and dictionary. Many of the constraints are so specific and only used in very special occasions that the application of linguistic control was postponed for later iterations. Instead, the main focus was implementing the visual representation of the database and how the Tokens, Arcs and their placement on the map would be placed. This is described in the following sections.
5.2 Design

The goal of the prototype is to allow users to create Arc Maps by creating Tokens and Arcs. To do this, several features must be implemented, which can be categorized within User Interaction and ArcMap representation.

Within user, interactions are considerations into how the user will interact with the software to create maps, and in ArcMap representation, we will describe the considerations made for the application to serve as a representational tool for ArcForm.

5.2.1 User Interaction

The design of the interface will be based on the design requirements established in sections 3.3.1. In this initial prototype, we will focus on establishing basic usability functionality in regards to user interaction and map creation.

The prototype had a series of design guidelines. As with most other representational applications, like The Brain, Lucidchart, or Xmind, there are some basic industry standards when selecting what each keyboard & mouse interaction should perform for kind of action. Selecting the correct keys and clicks certainly depends on the intended target group, context and intended action; however, many actions are often mapped to most used interactions within an application. With this mindset, creating Tokens, Arcs, and dragging and dropping them on top each other, was deemed the main type of action required by the program. The way to navigate, click and control the prototype was therefore primarily based on tried-and-true key-mapping with affordance in mind.

Based on this, we decided that the user interaction will be based on using the mouse and keyboard to place, label and connect Tokens and Arcs.

5.2.2 ArcForm representation

For the application to function as a representational tool for Arc from it must in it’s the most basic form to represent the Token and Arc combinations described in section 3.3.

As the representational artefacts created by the tool will vary in size and structure depending on the needs of the user. The visual representation of Arc maps must be able to show this. Based on the description of ArcForm, the Tokens can be represented as static objects at different locations with a text label denoting their subject. The Arcs, on the other hand, must be to dynamically vary which direction they are pointing to and from, as well as be able to change in size and length depending on the distance between the Tokens it connects.

Additionally, for a user to have the ability to create personalized ArcMaps, they must be able to label and relabel, Tokens and Arcs.
Based on this, the two following requirements of the application can be deduced.

- The user must be able to place and connect Tokens with mouse and keyboard input
- The user must be able to label Tokens and Arcs

These requirements are further expanded to include the following:

- The user must be able to place and name Tokens.
- The user must be able to connect Tokens with Arcs.
- The user must be able to connect Arcs to Arcs.
- The user must be able to label Arcs.
- The user must be able to label Tokens.

In the following section, we describe the decisions made in the implementation of these features.

5.3 Implementation

Based on the design, it is clear the application requires four main features to be able to support the design requirements.

For a user to create an Arc map, the basic features must be implemented.

- Placement of a Token
- Connecting Tokens with Arcs
- Labelling Tokens
- Labelling Arcs

In the following sections, we will discuss how these individual features were implemented, and within a description of how the various classes work.

In unity, all objects which are rendered and moved in space are associated with Rendering & Transformation Component. This means that every object which needs to be rendered on screen has a Renderer component, and every Object which has a location has a Transform Component.

Within the Unity Engine, there exists a multitude of Components that can give objects Physics and Collision behaviour.

For example, the Unitoken Object we decided to use a SpriteRenderer component to visualize the Unitoken with a .png image. We used a TextGUI Component
to render the text for the label of the Token; the transform component is used to translate the position of the object in space and finally, the Unitoken Class which we develop is used to manage the Arc Map Logic required from a Token.

In the rest of this implementation, we will use various Unity Components to visualize and translate objects, but we will focus on describing the Classes developed to create Arc Maps.

### 5.3.1 Map Creation

To create an ArcMap is must be possible to create an ArcForm Token. In the following section, we will describe the various classes implemented to be able to place Tokens on the map and link Tokens with Arcs.

#### Unitoken

The Unitoken class by itself is very simple, as the Tokens need to be visualized by a label and a position in space, the visualization of a Token can be seen in Figure 5.1. The method for initializing the Unitoken class can be seen in Listing 5.1.

```java
1 public void Initialize(string label, Vector3 transientPosition) {
2     myLabel.text = label;
3     this.TransientPosition = transientPosition;
4 }
```

Listing 5.1: Initialize Method

#### Arc

The Arcs within the system are meant to dynamically scale in length depending on the distance from source and target. Furthermore, to understand the direction in which an Arc is pointing, an arrow was necessary.
In the implementation of the Arc, the LineRenderer Component was used. The LineRenderer is a Unity component which renders a line based on an array of vectors. The first vector is the start position, and the last vector is the end.

The main logic behind the Arc is occurring with the method RefreshArc in the Arc class; the method can be seen in listing 5.2.

```csharp
public void RefreshArc()
{
    Vector3[] points = new Vector3[2];
    points[0] = source.transform.position;
    points[1] = target.transform.position;

    // Calculate the vector which points from the source to the target
    Vector3 sourceToTarget = (points[1] - points[0]).normalized;
    points[0] += sourceToTarget * ArcMapManager.Instance.linePadding;
    points[1] -= sourceToTarget * ArcMapManager.Instance.linePadding;

    center = points[1] + (points[0] - points[1]) / 2.0f;
    // Update the positions of the linerenderer component
    myLine.SetPositions(points);
    // Refreshes the collision collider
    UpdateCollider(points);
    SetTransientPosition();
    // Move the label to the center of the Arc
    UpdateLabelPosition(center);
    // Update Arrowsprite rotate the arrow to point towards the target
    UpdateArrowSprite(points[1], sourceToTarget);
}
```

Listing 5.2: RefreshArc Method

The resulting Arc can be seen in figure 5.2.

5.3.2 Connecting Tokens with Arcs

To allow users to connect Arc to one another, first, there must be a system in place which can create Tokens and Arcs.

Factories

The purpose of the factories is to develop an abstraction class which takes of everything associated with instantiating objects of specific types.
In the implementation, we use two different factories, the Token factory and the Arc factory. The methods for the Token factory quite create a Token with at the position input and labelled with the label input:

```
public Unitoken AddNewToken(string Label, Vector3 position){
    Unitoken newToken = Instantiate(unitTokenPrefab, position, Quaternion.identity, transform.parent).GetComponent<Unitoken>();
    // Initialize
    newToken.Initialize(Label, position);
    // Add to list of token
    ArcMapManager.Instance.AddTokenToList(newToken);
    return newToken;
}
```

Listing 5.3: AddNewToken method

The Arc collection factories AddNewArc method is quite similar, instead of receiving a position vector, it receives two Tokens, a source and a target. The source and target Tokens are the same which are used in listing to determine the path of the Arc. An example of the resulting combination of Tokens and Arcs can be seen in figure:

```
public Arc AddNewArc(Unitoken Source, string label, Unitoken Target)
{
    Arc arc = Instantiate(joinArcPrefab, Vector3.zero, Quaternion.identity, transform.parent).GetComponent<Arc>();
    arc.Initialize(source, target);
    arc.SetLabel(label);
    ArcMapManager.Instance.AddArcToList(newJoinArc);
    return arc;
}
```

Listing 5.4: AddNewArc Method
5.3. Implementation

Mouse Listener

Now that we can create Arcs, the application now needs to support Token and Arc creation through user interaction. To do this, we implemented the Mouselistener class as an interface for registering user input.

For this implementation, the following functionality was implemented in the Mouselistener.

- Drag from Background to Token
- Drag from Token to Background
- Drag from Background to Background

All of this is implemented through the use of the three following methods, `OnDragFromBackground`, `OnDragFromToken` and `OnDraggedRelease`.

The `OnDragFrom` methods check if the mouse has been clicked on either the background or a Token and then determines if the mouse is currently being dragged. The `OnDraggedRelease` method then determines if the mouse was released over a Token or the background. If the mouse is released from dragging over a Token, an Arc which points from the starting position of the dragging event to the Token is created. If the mouse is released on the background, then a completely new Arc is created with Tokens at the beginning of the dragging event and a Token at the end. Finally, if the dragging starts from a Token and ends on the background or a Token, then a new Arc is created from the source Token, either to the Token at which the dragging has ended, or a new Token is created at the mouse position.

5.3.3 Labelling

To implement the functionality of labelling Arcs and Tokens, the structure of the system was reconsidered. As both Arcs and Tokens were capable of being labelled,
it is quite straightforward to imagine that the labelling method for each could be shared.

This is accomplished through polymorphism, as a parent class to both Arcs and Unitokens were developed, which we call the Fragment class. The fragment class contains the logic required to respond to user input and bring forth a text input field. The text fields content is then saved as the label of the fragment.

5.4 Evaluation

In this section we will describe how we evaluated the first iteration of the application, the main purpose of this evaluation is to discover usability issues, which will be determined through the use of observations and interviews.

The evaluation of the application in this stage was performed through expert interviews, experienced usability designers, as well as with the creator of ArcForm. The expert interviews were conducted as informal conversational interviews, and then the main points of discussion were the applicability and usability of the application as well as an attempt to determine which features could be automated by the system to make it easier to create ArcMaps.

To find out how participants interacted with the application. The findings were intended to help us determine whether specific tasks need to be developed or displayed differently. Our definition of usability suggests that the application should be: efficient, effective, easy to learn, easy to remember, and have good utility (i.e. good usability). It should also support creativity and be motivating, helpful, and satisfying to use (i.e. good user experience). The application is designed for researchers, so the range of users is broad in terms of age and experience with technology. The tasks are a small sample of the entire set prepared by the test conductors. They cover adding uni-Tokens, navigating Token relationship, searching the database, and finding a specific connection – which are common activities within this sort of notation tool.

5.4.1 Test steps

1. Introduction to test and purpose.
2. Consent form and signing.
3. Screen recording enabled.
4. Basic introduction to the application.
5. Participant number filled in the application.
6. Start test 1 with nine tasks.
5.5. Results

7. Post-test interview.

8. Exit screen capture. Task times are logged automatically to a .txt file while running.

5.4.2 Test Setup

A suitable laptop computer was selected to run the application, with the following specifications: MSI gaming computer, i7-6700hq CPU, 16 GB ram, Nvidia GTX 970m.

List of hardware:

- MSI gaming computer.
- Logitech MX master 2s wireless mouse
- 27 inch Dell screen model SE2717

The test was conducted in an office space with minor background noise. The test computer was set up on its table with a room separator to divide the test participant and researcher. An external monitor was used so the researchers could observe each participant interact with the application freely, without interruptions or pressure. The external screen was situated on the opposite side of the room separator.

5.5 Results

The results of this iteration are determined through an examination of interview notes and suggestions generated from the expert interviews.

5.5.1 Interview Results

Several main points of interest which we understood from the interviews were there were functionality errors which distracted from the purpose of the application.

Feedback was that the application itself did not separate itself from a mind-map

5.6 Conclusion

Based on the results of the evaluation, we feel confident in expanding the functionality of the software to be able to investigate how we can implement features which encourage relational reasoning and creativity. How we plan on doing this will be discussed in the next chapter.
Chapter 6

Iteration 2

6.1 Introduction

In the first iteration, we developed a simple application where the user could place Tokens and Arcs; this resulted in an application capable of creating simple ArcMaps.

We discovered that for the application to support relational reasoning and creativity, additional functionality must be designed and implemented.

In section 3.1.3, we discuss how creativity can be influenced by related information and in 3.2, we discuss how relational reasoning is an ability which allows us to discern patterns in information.

Based on the design requirements in section 3.3.1, we will now investigate how to influence creativity and relation reasoning in the application.

6.2 Re-Design

In this section, we will describe how we will change the software design to meet the new requirements.

To influence creativity and relational reasoning, we believe that the application must be able to provide the users with related peripheral information to a subject they are working with. Furthermore, we believe that allowing the user to establish relationships by discerning patterns in these associations can help support relational reasoning.

Therefore a requirement to the implementation of the software is to enable the application to systematically query a database for related topics and terms related to a Token or Arc which the user is working with.

From this requirement, the following implementation requirements can be extrapolated.
Chapter 6. Iteration 2

- The system must be able to query and retrieve related information about a subject
- The system must be able to generate new Arc and Token based on these relations
- The user must be able to select and de-select relations which they are interested in

Based on these requirements, we decided to use The ConceptNet Database for gathering related terms.

The ConceptNet database is a Semantic Network Database created to help machine learning algorithms understand the relationships between concepts. A user of the concept net can query the database for any topic, and it will return semantic relations from 37 categories. For example, if "Dog" is queried, information such as "has four legs", "barks", "animal", "canine" etc is returned.

For the user to be able to interact with the results returned from the ConceptNet, we decide that a dynamic menu system which changes in size and content depending on the response would be an intuitive interface for selecting and de-selecting Tokens.

Finally, we assume that the Tokens created from interacting with the menu must follow some system which chooses where it is placed on the map. The system we decided upon is a graph layout system based upon the Force-Directed graph approach. A force-directed graph is a way of configuring the layout of a graph by allowing an algorithm to move objects in the graph to avoid overlap automatically.

From this design consideration, the following implementation requirements are extrapolated.

- The application must use ConceptNet to retrieve related terms for the user.
- The terms must be presented within a Dynamic Menu
- The Tokens created from interacting with the menu will be automatically placed through force-directed graph system

6.3 Implementation

In this sections the implementation of the var

6.3.1 ConceptNet & DBPedia integration

As we wanted the software to allow users to search for specific terms, and then use related and associated terms to create Arc maps, an integration of the application into Knowledge Database as well as a Semantic Network was required.
6.3. Implementation

The following sections will describe the expected workflow of a user and then the implementations we performed to support it.

In the first stage of using the application, the user will search for a subject they are interested in. Therefore the SearchEngine class was developed as an interface between the application and Dbpedia. The purpose of the SearchEngine class is to receive a query in the form of for example "Dog" ask DBpedia for any results with that label (or related to), and return a list of recommendations.

By querying the DBPedia PrefixSearch API, it is also possible for the user to spell their query incorrectly and still receive associated results, as well as autocompletion functionality.

When the query has completed the results from the prefix search are returned as an XML file which is parsed into a C class. The results are displayed to the user, and when they select on a new Token with the name of their selection is created, an image of the search panel can be seen in figure 6.1.

The next stage of the application is the association/search with then uses the ConceptNet. When a user selects a Token, a query is sent to the ConceptNet database requesting any all relations which the database has in relation to the subject. The results from Concept are returned as JSON-LD objects which are then parsed in into C classes which can be used in unity.

6.3.2 Dynamic Menu

ConceptNet has up to 37 different semantic categories for any subject which the user should be able to select through we developed dynamic menu system. The system loads in every relation received from ConceptNet and nests the attributes within categories as can be seen in figure 6.2.

6.3.3 Force Directed Graph

To organize and structure the layout of the ArcMap, we decided you implemented a force-directed graph system. The implementation is based on the principle, which is that every Token and Arc must maintain a minimum distance from one another.

This is implemented in the ArcMapLayout Class, which contains the methods GetForceVectors and AddForces. The GetForceVectors compares the distances between the position of every fragment within the supplied array and returns a force vector for every fragment, which points away from every other fragment. The AddForces method simply adds the forces to the positions of the input fragments.

```
public Vector3[] GetForceVectors(List<Fragment> fragments)
{
    Vector3[] forces = new Vector3[fragments.Count];
    for (int i = 0; i < fragments.Count; i++)
    {
        Fragment frag = fragments[i];

        for (int j = 0; j < fragments.Count; j++)
        {
            Fragment neighbour = fragments[j];
            float distance = Vector3.Distance(frag.TransientPosition, neighbour.TransientPosition);
            if (distance > minDistance && distance < maxDistance)
            {
                Vector3 dir = (frag.TransientPosition - neighbour.TransientPosition)/distance;
                forces[i] += dir;
            }
        }
    }
    return forces;
}
```

```
public void AddGraphForces(List<Fragment> fragments)
{
    Vector3[] fragmentForces = GetForceVectors(fragments);
    for (int i = 0; i < fragmentForces.Length; i++)
    {
        fragments[i].transform.position += fragmentForces[i] * Time.deltaTime;
    }
}
```
6.3. Implementation

The final view of the application can be seen in figure 6.4.

6.3.4 Test Manager

To assist in the evaluation of the software, the Test Manager Class was implemented as a virtual test conductor. The purpose of this class is to present the user with different tasks which they must complete in the software. The tasks can be categorized into four types, and are based on the test requirements described in section 4.4.
Chapter 6. Iteration 2

The four task types are: Find Token, Isolate Token, Find Relation and Isolate Relation. The find Token and isolate relation tasks are quite simple in implementation as the system simply need to determine if a Token with the name specified in the tasks exists on the map.

The Isolate tasks are slightly more complicated as the task is completed when the Token specified in the task is the only one placed in a branch.

The test manager is constructed in such a way that every task can easily be created by simply entering the required parameters required to complete the task. An image of a test task can be seen in figure 6.5.

During the application run time, every time a task is completed the Test Manager will automatically display a panel with the description of the next task which the participant must complete an image of this in action, can be seen in figure 6.6.

6.4 Evaluation

In this section we will describe how we evaluated the first iteration of the application, the main purpose of this evaluation is to discover usability issues, which will be determined through the use of observations and interviews.

6.4.1 Test Setup

The test setup consisted of one laptop with the application running connected to an external screen and keyboard. The external screen and keyboard are separated
from the laptop with a room divider. The purpose of the external screen is to allow the test conductors to observe how the user interacts with the application with minimal interference. A screen recording of every participant is recorded for data analysis.

6.4.2 Test Procedure

The test was run by two test conductors, one responsible for introducing the application and answering questions, the other responsible for taking observation and interview notes.

The test participant is introduced to the purpose of the test, which is to test the usability of the application; they are not informed about the purpose of the application.

The participant is introduced to the various way of interacting with the application, and a list of interactions available is explained. This explanation takes place in a "sandbox" environment where the observer guides the participant through a variety of queries and interactions with the application. The procedure behind this is described stepwise below.

1. Ask the participant to search for "Human."
2. Ask the participant to select the human result
3. Ask the participant to use the menu to add and remove Tokens
4. Ask the participant to move the camera
5. Ask the participant to query a Token by selecting it with the mouse
6. Ask the participant to remove a branch of the Token with the mouse and with the menu

At this point, any questions by the participant are answered, and screen recording will be initiated.

The test consists of three different test scenarios, which each involve the participant completing nine tasks. Between each test, an interview is held and the questions can be seen in Appendix B.

6.5 Results

The results of the interview notes were coded into four categories to determine which features were important to focus on. The data is coded by counting statements which show sentiment in three categories. Interaction, Overview and Comprehension
Two participants found the interface intuitive. One participant found it difficult to interact with the menu. Four participants experienced a loss of overview. One participant found the tasks confusing.

I they moved around and constantly were in motion. I didn't like it. It was irritating that the nodes kept jiggle. It is a funny way of structuring the graph; it is placing the nodes very weirdly and without structure, which goes against my OCD.

The notes from the evaluation can be seen in Appendix C.

6.6 Conclusion

We believe the following quotes from the evaluation are an important basis for feature implementation and design re-considerations.

As four out of five participants commented negatively about the lack of overview in the application, we believe that a redesign focused on that should be followed. Furthermore, an additional list of requirements based on observations was established.

Problem Areas

- UI Feedback Inconsistent
- Loss of overview

Features

- Move map with middle mouse click and drag
- Re-design map layout system

Bugs

- Refresh menu when removing Tokens
- Inconsistent feedback when selecting and deselecting Tokens from menu

We conclude from the test interviews, that the base usability of application is at a level where the frustration from the participants feel while working with the application stem less from the interaction with the interface and more with the actual content, this leads us to believe that in the following iteration a focus on re-designing the automatic layout of the Arc maps and fixing the menu issues is paramount.
Chapter 7

Iteration 3

7.1 Introduction

The focus of this iteration is to fix usability errors, which were observed during the previous evaluation.

Based on the results of the previous iteration, we decided to re-design specific features of the application.

7.2 Re-Design

Based on the results of the previous application, in this iteration, there was a focus of increasing the usability of the system further.

It was determined that a loss of overview was prevalent in the previous iteration, which this re-design will focus on fixing.

Based on the results of the interviews and observations, it is clear that the force directed graph works against the users’ ability to maintain an overview of the ArcMap during construction. Therefore a system which places the Tokens automatically in an according to the grid rather than in a random fashion seems to be the clear alternative.

To further attenuate the differences between the relationships within the menu and on the map, we decided that a colour coding system should be implemented. The purpose of this system is to allow users to scan the ArcMap for specific colours to discern relationships.

Finally, an issue which was determined in the previous iteration, which caused users frustrating, was the inconsistent feedback provided from the menu when selecting and de-selecting Tokens.

Therefore based on these statements, the following requirements can be extrapolated.

• A grid-based system to organize the layout of the Arc map
• Color coding of different relationships which influence the menu, Tokens and Arcs

7.3 Implementation

7.3.1 Grid based Layout

The implementation of the grid-based layout is implemented in the ArcMapGrid Class.

This class has a single method the find empty spot method which uses a separate class called the Searcher who is responsible for finding an unoccupied spot in the grid for a new Token to fill.

```csharp
public List<GridCell> FindEmptySpot(Vector3 position, int targetSize, out Vector3 outPos)
{
    Searcher searcher = new Searcher();
    searcher.size = targetSize;
    searcher.startPos = position;
    searcher.currentPos = searcher.startPos;
    searcher.startDirection = new Vector3(1, 1, 0);
    while (searcher.searching && searcher.searchCount < 10)
    {
        searcher.Run();
    }
    outPos = searcher.endPos;
    return searcher.GridCells;
}
```

Listing 7.1: FindEmptySpot method

The searcher works by first receiving a starting position where it will look for empty cells surround it, the search movement of the searcher is illustrated in figure 7.1. If it cannot find an empty spot initially, it will expand the range of the search, until a spot is found.

```csharp
public void Roam()
{
    if (checkedLocations == null)
    {
        checkedLocations = new List<Vector2Int>();
    }
    for (int i = 0; i < searchPattern.Length; i++)
    {
        for (int j = 0; j < searchCount; j++)
        {
            currentPos = startPos + (searchPattern[i % searchPattern.Length] + searchPattern[(i + 1) % searchPattern.Length]);
        }
    }
}
```

7.3. Implementation

With this, the placement of the Tokens in the grid becomes far more predictable and structured. An example of this can be seen in figure 7.2.

7.3.2 Color Coding

The colour coding of the application was designed through the implementation of a ColorToken structure; the ColorToken struct contains three different colours for every single ConceptNet relation. The three colours represent the colours for the menu buttons interactions states. There are three interaction states which are, when the mouse hovers over the button, when the button is selected and when the button default state. The same colour states for the buttons are used for the Tokens to establish a visual identity for the category.
The colour coding influence on the visual style can be seen in figure 7.3 and figure 7.5 and finally, the visual style of the application can be seen in figure 7.6.

7.4 Evaluation

The second iteration test evaluates the performance of the application with participant comments for a more in-depth explanation to specific actions, good or bad concerning efficiency. The second iteration varies from the first in terms of content, design and tasks. The complete task setup for all three scenarios can be found in the appendix A.

7.4.1 Test steps

To measure the performance of the application and gather more data, the test is evaluating the three different scenarios in the second iteration. The test setup is the
7.4. Evaluation

Figure 7.4: Updated Token Style

Figure 7.5: Updated Thought Style

Figure 7.6: Updated Visual Style
Chapter 7. Iteration 3

same as in the first iteration, but a different procedure is used, which is as follows:

1. Introduction to test and purpose.
2. Consent form and signing.
3. Screen recording enabled.
4. Basic introduction to the application.
5. Participant number filled in the application.
6. Start test 1 with nine tasks.
7. Post-test 1 interview (interview details link)
8. Continue to test 2 with nine tasks.
9. Post-test 2 interview (interview details link)
10. Continue to test 3 with nine tasks.
11. Post-test 3 interview (interview details link)
12. Exit screen capture. Task times are logged automatically to a .txt file while running.

List of questions:

1. What are your first impressions of the application? (Only after the first scenario)
2. Have your impressions changed? (Only second and third)
3. How did that make you feel? - Why?
4. What do you think the purpose of the software is? - Why? (Only after the third scenario)
5. Do you have any questions for us regarding the test?

7.5 Results

The notes from the evaluation can be seen in Appendix C

Based on the analysis of the observation and interview notes, the following results core statements in regards to the applications

The results of the interview notes were coded into four categories to determine which features were important to focus on. The data is coded by counting
7.6. Conclusion

Based on the observations and the interview notes we conclude the following:

- The application is intuitive and easy to use.
- The tasks are difficult to understand.
- The relationships are illogical, making it frustrating to complete some of the tasks.

Based on the results of the test, we developed a list of features which could improve the usability of the application further. Problem Areas:

- ConceptNet Illogical Relations

Features:

- The ability to select and move Tokens
- The ability to create an unlabeled Token for personal labelling
- Menu which appears at the location of the Token
- ShortCut to focus a Token to the centre of the screen
- Remove ConceptNet query Limit
- Collapse and Concatenate Collections of Arcs
- Auto Unitokenality
- Menu Search Filter
- Local database to contain custom relations

Specific Errors
• Refresh menu when removing Tokens
• Inconsistent feedback when selecting and deselecting Tokens from the menu

We conclude from the test interviews that the base usability of the application is at a level where the frustration our participants feel while working with the application, stem less from the interaction with the interface and more with the actual content. This leads us to believe that in the following iteration a focus on re-designing how the application queries concept net, as well as evolving the evaluation to test the usefulness of the system in regards to promoting relational reasoning and creativity.
Chapter 8

Discussion

In this section, we will discuss the various considerations we have made concerning the scope of the project, the goals of the application and then the purpose of the evaluation.

8.1 Project direction

The initial project focus was in investigating what sort of application could be implemented to assist researchers during the research phase of a project; this was heavily motivated from personal experience of developing many different projects, in which a consistent dread and annoyance with the tools available for maintaining overview and structure.

It could indeed be argued that the project could have focused on developing a representational tool based on ArcForm, or on developing unique representational notation extrinsically linked to an external database, or we could have focused on how specifically relational reasoning could be supported.

We believe that with an intuitive interface, in future re-designs, it would be possible to map complicated scenarios which would allow us to test for the previous hypotheses.

Unfortunately, the implementation of the software and the consistent usability issues meant that an evaluation which tested the impact on relational reasoning and creativity was never accomplished. The advantage of this inspired us, on the other hand, to experiment with many different ideas and features which were implemented and removed, with only the most significant changes described in the three iterations.

We believe that the theories described in the analysis support the potential of the idea, and we are confident that further refinements and changes to the presentation, the database queries and the interface could lead to very versatile application.
8.2 Evaluation

The purpose of the evaluation was mostly to focus on the basic usability of the application as our hypothesis stood that, unless the basic usability of a system is in order, then no amount of new functionality can be explored by the users.

Many of our decisions in regards to which features would be implemented were based on informal interviews and observations carried out outside of rigid test environments, as well as personal evaluations. This allowed to switch perspectives and designs quite quickly, but also hamstrung our ability to conduct any focused evaluation.
Chapter 9

Conclusion

The purpose of this project was to investigate how an application created to create a representational artefact for the purpose of knowledge mapping, could be created to support relational reasoning and creativity.

Our findings show that many participants realised the purpose of the application without any description of its purpose, yet to unequivocally state the representational tools has any influence on creativity, an evaluation designed to test for this specifically must be performed.

This type of test should be performed in concert with a parallel project to determine the specific needs of researches during the different research phases.

We conclude, that to reach the applications and our goals, a rigorous iterative methodology must be applied to refine further and add additional functionality.
Chapter 10

Future Works

Future works with the application would focus on investigating how to lift the representations from within the arc maps from simple word relations for example illustrated in figure 10.1.

The creation of nested statements would allow more complicated maps to queried from a user perspective, and investigation into how the system could store and retrieve information from a database. Furthermore, the application could be extended with functionality allowing hyperlinks, images and videos to be connected to thoughts, allowing the software to fully supporting information gathering. Finally the application would be able to allow users to compare arcmaps, and rate thoughts based on the sources and arguments.

Figure 10.1: Arc Form map from Allsop [2]
Bibliography


Appendix A

2nd & 3rd Iteration Scenarios

The tabular below displays the first scenario task list used when evaluating the application. Unitoken is abbreviated U. Collection Token abbreviated CT.

Table A.1: Scenario 1 task list

<table>
<thead>
<tr>
<th>Task no.</th>
<th>Task instruction</th>
<th>Success criteria</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Create a “Dog” token.</td>
<td>Dog (U)</td>
<td>Search the Database</td>
</tr>
<tr>
<td>2</td>
<td>What are a Dog’s capabilities?</td>
<td>Capable of (CT)</td>
<td>Use the Collection Menu</td>
</tr>
<tr>
<td>3</td>
<td>Edit your graph so it only has “run” as a capability.</td>
<td>Run (U)</td>
<td>Navigate the Collection menu</td>
</tr>
<tr>
<td>4</td>
<td>Add the “is a” relationship for the “Dog” token.</td>
<td>Is a (CT)</td>
<td>Understand what a Collection Item is as opposed to Unitokens</td>
</tr>
<tr>
<td>5</td>
<td>Select “pet”.</td>
<td>Pet (U)</td>
<td>Navigate the map, Change Unitoken focus</td>
</tr>
<tr>
<td>6</td>
<td>Add the “related to” relationship and select “Animal”.</td>
<td>Animal (U)</td>
<td>Understand building a network springs off Unitokens</td>
</tr>
<tr>
<td>7</td>
<td>Find out what an animal “desires”.</td>
<td>Desires (CT)</td>
<td>Move the map with zoom + controls</td>
</tr>
</tbody>
</table>
Continuation of Scenario 1 task list

<table>
<thead>
<tr>
<th>Task no.</th>
<th>Task instruction</th>
<th>Success criteria</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>Remove everything but “Dog”.</td>
<td>Capable of (CT), Is a (CT)</td>
<td>Understand deletion and how to do so.</td>
</tr>
<tr>
<td>9</td>
<td>Find out how “Cat” is different from a “Dog”.</td>
<td>Moderator ends test</td>
<td>Test of application knowledge so far. All interactions are used in order to complete the task successful. Time logging, Error detection count vs minimal usage.</td>
</tr>
</tbody>
</table>

The tabular below displays the second scenario task list.

**Table A.2: Scenario 2 task list**

<table>
<thead>
<tr>
<th>Task no.</th>
<th>Task instruction</th>
<th>Success criteria</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Create a “Cult” token.</td>
<td>Cult (U)</td>
<td>Search the Database</td>
</tr>
<tr>
<td>2</td>
<td>What is a “Cult” related to?</td>
<td>Is related to (CT)</td>
<td>Use the Collection Menu</td>
</tr>
<tr>
<td>3</td>
<td>Edit your graph so it only has “Loyal” as a relationship.</td>
<td>Loyal (U)</td>
<td>Navigate the Collection menu</td>
</tr>
<tr>
<td>4</td>
<td>Add the “similar to” relationship for the “Loyal” token.</td>
<td>Similar (CT)</td>
<td>Understand what a Collection Item is as opposed to Unitokens</td>
</tr>
<tr>
<td>5</td>
<td>Select “Doglike”.</td>
<td>Doglike (U)</td>
<td>Navigate the map, Change Unitoken focus</td>
</tr>
<tr>
<td>6</td>
<td>Add the “is related to” relationship and select “Dog”.</td>
<td>Dog (U)</td>
<td>Understand building a network springs off Unitokens</td>
</tr>
<tr>
<td>7</td>
<td>Find out what a Dog “desires”.</td>
<td>Desires (CT)</td>
<td>Move the map with zoom + controls</td>
</tr>
</tbody>
</table>
Continuation of Scenario 2 task list

<table>
<thead>
<tr>
<th>Task no.</th>
<th>Task instruction</th>
<th>Success criteria</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>Remove everything but “Cult”.</td>
<td>Capable of (CT), Is a (CT)</td>
<td>Understand deletion and how to do so.</td>
</tr>
<tr>
<td>9</td>
<td>Create a network where “Cult” leads to “Church”</td>
<td>Church (U)</td>
<td>Test of application knowledge so far. All interactions are used in order to complete the task successful. Time logging, Error detection count vs minimal usage.</td>
</tr>
</tbody>
</table>

The tabular below displays the third and final scenario task list.

**Table A.3: Scenario 3 task list**

<table>
<thead>
<tr>
<th>Task no.</th>
<th>Task instruction</th>
<th>Success criteria</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Create a “Arson” token.</td>
<td>Arson (U)</td>
<td>Search the Database</td>
</tr>
<tr>
<td>2</td>
<td>What is a “Arson” related to?</td>
<td>Related to (CT)</td>
<td>Use the Collection Menu</td>
</tr>
<tr>
<td>3</td>
<td>Edit your graph so it only has “incendiary” as a relationship.</td>
<td>Incendiary (U)</td>
<td>Navigate the Collection menu</td>
</tr>
<tr>
<td>4</td>
<td>Add the “is related to” relationship for the “incendiary” token.</td>
<td>Is related to (CT)</td>
<td>Understand what a Collection Item is as opposed to Unitokens</td>
</tr>
<tr>
<td>5</td>
<td>Select “arsonist”.</td>
<td>Arsonist (U)</td>
<td>Navigate the map, Change Unitoken focus</td>
</tr>
<tr>
<td>6</td>
<td>Add the “capable of” relationship and select “burn a building”.</td>
<td>Burn a Building (U)</td>
<td>Understand building a network springs off Unitokens</td>
</tr>
<tr>
<td>7</td>
<td>Select “arsonist”, add the “is related to” relation and select “act”.</td>
<td>Act (CT)</td>
<td>Move the map with zoom + controls</td>
</tr>
<tr>
<td>Task no.</td>
<td>Task instruction</td>
<td>Success criteria</td>
<td>Purpose</td>
</tr>
<tr>
<td>----------</td>
<td>------------------</td>
<td>------------------</td>
<td>---------</td>
</tr>
<tr>
<td>8</td>
<td>Remove everything but Arson.</td>
<td>Arson (U)</td>
<td>Understand deletion and how to do so.</td>
</tr>
<tr>
<td>9</td>
<td>Create a network where “Arson” leads to “crying is part of life”</td>
<td>Life (U)</td>
<td>Test of application knowledge so far. All interactions are used in order to complete the task successful. Time logging, Error detection count vs minimal usage.</td>
</tr>
</tbody>
</table>
Appendix B

2nd & 3rd Iteration Evaluation Questions

The moderator conducted a semi structured interview in between every evaluation scenario. After the final scenario the moderator asks some general questions. The interview questions were as follows:

Questions asked in between every evaluation scenario

1. What are your first impressions of the application?
2. Have your impressions changed?
3. How did that make you feel?
4. Why?

Questions asked after the third scenario

5. What do you think the purpose of the software is?
6. Why?
7. Do you have any questions for us regarding the test?
Appendix C

2nd & 3rd Iteration Participant Responses

C.1 2nd Iteration interview responses

The following sections contain each of the participants that evaluated during the 3rd iteration.

C.1.1 Participant 1

1st interview
I found the objects were hard to click since they moved around and constantly were in motion. Sometimes it was hard to locate the objects I just created. It was actually a little annoying since I constantly lost the task I had to do.

2nd interview
It was hard to complete the last task, every time I spawned in one direction it was not so easy to orientate. My impressions have not changed, I don't really know what I should use this for.

3rd interview
Same as before, The map moves a lot around on its own. (Autofocus on new Unitoken). I kept losing track on the direction I was heading. It might be some kind of mind map? I think it is because the objects have connection to each other. No questions.
C.1.2 Participant 2

1st interview
Once I found out what the program did it was kind of intuitive to use. Selecting things from the menu put it on the map. Pretty basic but I liked how I could select things from the menu. I wish I could interrupt the focus when it moves the map. It felt a little time wasting that I couldn’t do any movement while it did so.

2nd interview
Yes (my impressions have changed), this time the graph felt a bit unstable. It was very hard to complete the last task since the map was all over the place whenever I spawned tokens through the menu. I wish I could just search the goal right away (task 9).

3rd interview
The last task was easier. - I think I understand the app better now. Maybe it, i.e. the program, it is a helper to think differently or to see connection between things. There are arrows pointing to other objects which sometimes share an understanding.

C.1.3 Participant 3

1st interview
The menu seemed quite nice, It was easy to move around and select objects and create other relations. The tasks were easy, but I don’t really know what the purpose is with it? It kind of made me want to know what it is used for.

2nd interview
No (no new impressions), it still responds quickly to what I click on. I guess it made me happy to complete the tasks fast.

3rd interview
Wow, it didn’t really make sense that cry and crying was related in that sense (refers to scenario 9). It seemed a bit far fetched. That confused me a bit and it was pure trial and error. It is probably a map to display connections or so maybe for a database.
C.1.4  Participant 4

1st interview

I didn’t like it. It was irritating that the nodes kept jiggle (adjusting to each other e.g. forced directed graph) and it was hard to get an overview when everything was folded out.

2nd interview

I think it is a bit better now when i kinda see what it is used for. I am used to moving with the mouse on the canvas, not using WASD (standard gaming controls on the keyboard). I think it would help a lot if I could move by dragging the background around instead.

3rd interview

It is more or less the same. Well the map still acts the same and the tasks are the same. I think the purpose is to understand different dependencies between names. Use it like a mind map.

C.1.5  Participant 5

1st interview

It is a funny way of structuring the graph, it is placing the nodes very weirdly and without structure, which goes against my OCD. I expected the nodes to be placed somewhere near the origin point but it seems like it explodes around. I feel slightly annoyed since I like a neat and pleasant layout

2nd interview

I actually quite like it now. It is nice how it creates a path for me in an expansive way. It might be because i’m getting used to the system that I find it better than before.

3rd interview

Well, it has its kind of obscure placement, but for some reason i’m getting used to work in the chaos. It is some collection of knowledge and its relationship. It derive that from how it works. How its relations are set up and how the words always compare to each other.
Appendix C. 2nd & 3rd Iteration Participant Responses

C.2 3rd Iteration interview responses

The following sections contain each of the participants that evaluated during the 3rd iteration.

C.2.1 Participant 1

1st interview

Annoying menu, moving focus away from operating area. Not nice that you cannot move already placed tokens around. It is quite simple to work with. I see how you can use it as a mind map. Menu should be where the Unitoken is.

2nd interview

Difficult to accomplish the last task, I found it difficult to find the correct relation I looked for in my understanding/logical. Frustrating to try many different themes to find/accomplish the task, normally I would search for the correct combination. Or search for the word “church”.

3rd interview

Much easier task to work from (task 9). I feel I’m informed to find very stringent goal. I think this is a type of association map to create a cluster of knowledge to inform some directions I maybe would look into which was not occurring to me, different perspective.

C.2.2 Participant 2

1st interview

First impression: I liked it; I was missing, zoom in or out on keyboard; collapse/minimize tokens. Situation: If I was not sure if I needed a specific token, id rather just minimize instead of removing the entire stream of thoughts. Happy – Intuitive, it used interaction that I already knew from other systems.

2nd interview

Tasks 9: Confusing. I think the tabs didn’t work for me, illogical branching. I wish there was a shortcut on the Keyboard “f” for focus. I thought the use was a bit more difficult: I think either it’s the wording or branching mechanism didn’t display the correct link from my mind.
3rd interview

I want "cry" to lead to "part of life". Please! Unitokenality - There were multiple objects with the same name. Interaction is still fine! Branching and grouping of database is quite bad. But the menu is nice and I like the interaction. I think the app is used for Mind mapping, Brainstorming? – that would be my answer. It is not a google search, but it is a visualizing way of displaying general thoughts.

C.2.3 Participant 3

1st interview

I don’t know the meaning of the software, so I found it hard to use. I didn’t understand the top menu (collection menu). I understood what the category meant after using it 3 times (within scenario 1). Began to act from memory.

2nd interview

No new opinions of the software, I still find it hard to use and search the menus. Still not fully content with its menu functionality.

3rd interview

More confusing than before. The question “9”, confused me a lot. Frustrating. I could think of Brainstorming as a way to use it. Based on: The setup, the visual style, branching.

C.2.4 Participant 4

1st interview

Functionality is quite good, and it made sense to navigate in the software. Menu overview was nice, visual linking and its chart was quite good. Happy, great overview.

2nd interview

It was weird I couldn’t go from Christianity to church. But overall the application makes sense and I can track myself going backwards again. That was actually really good (in relation to previous sentence).

3rd interview

The menu was close to being succumbed due to many options. The data base is kind of failing. Long searches on wiki can help me backtrack search pattern.
C.2.5 Participant 5

1st interview
Very intuitive, Expansive leads to a rabbit hole. Many branches, probably you can get lost quickly. As long as there is not too much of a network going on---> maybe if you could Patch it (referring to PureData and its ability to collapse items within itself like nesting)

2nd interview
Yes (my opinion has changed), it was a bit weird to look for church. It quickly became a long chain of selections.

3rd interview
Cry and Crying is the same for me.. So I found it confusing that "Crying" lead to the answer and not "Cry". I believe it is used as Mind Map, a Brainstorm thingy. Maybe it is used as Relationship, like in game design or lead design. Create a network of mechanics, situations or other. Game design makes a lot of sense, normally I use a notebook, but it is very bad for keeping the overview.

C.2.6 Participant 6

1st interview
Quite easy, Intuitive. Easy to pick descriptions of things from the menu. It worked.

2nd interview
More tasks, but I had to think how to get there, and it made sense along the way.

3rd interview
No different impression. I think it is used for designing something or how we assume relationships.

C.2.7 Participant 7

1st interview
Very intuitive. (Why) Well, the buttons did what I thought it would. It was easy to follow the tasks.
2nd interview

Still intuitive, but I had to take a few more paths to arrive at the end result. I think "The church" should have appeared within "Christianity". Expected a shorter path. I was pretty close to asking for help, which was a bit frustrating.

3rd interview

Same as it has been. Still intuitive, but the paths in order to complete was a bit difficult and misleading. A bit frustrating. Crying was a human activity, who would have thought? (Purpose) Probable something about AI (Artificial Intelligence), logic chain, making inferences.