Bringing Knowledge Through AI and SMS

Sam Heather
Department of Computer Science,
The University of York,
`sam@heather.sh`

April 7, 2015

Abstract

In remote Africa there are millions of disadvantaged and uneducated individuals, who in the vast majority do not have access to the internet the knowledge that this brings. Outside of their immediate friends and family, individuals can not access the information they need to further their education, or just for general interest.

In many parts of the world, the number of people without access to the internet but access to a mobile phone is significant. This project aims to research and develop a system capable of bringing knowledge through a question and answer based interactive system, in the language natively spoken by the user, through the use of a simple Artificial Intelligence and an SMS interface. The system will be expandable, such that it can be adjusted to handle questions on any knowledge area.

This project raises ethical issues relating to the responsibility of providing accurate information when in a position of trust, the ethics of machine translation and maintaining user privacy.

It is early days so it is normal that many things are missing from the abstract. You should keep in mind that it should include later on few sentences on method/experiment, result and conclusion.

Contents

List of Figures

List of Tables

Statement of Ethics

Pretty much the most important part - write this! - data protection - users information irreversibly obscured and protected in the database to protect them in case of database hack. - data protection - supervisor had root password to the server. - experiment - participants were asked to sign consent forms, these were stored with supervisor. - after experient, all participants were anonymised by a unique participant id, and the only mapping of these to names is the consent form, which the supervisor has - ethics of providing information - steps taken to try and prevent wrong info been returned

Acknowledgement

The author would like to acknowledge the following individuals for their contribution to this project:

1 Useful thoughts from Lilian

Take a quick look at list of other students previous projects to get an idea of how intro/lit review is structured etc.

Remember to take notes on when I come across problems, for my conclusion.

2 Introduction

2.1 Background of this project

The inspiration for this project originates from a project the author undertook in September 2014 whilst attending Yacht Hack, a week-long hackathon on a Yacht in Croatia[1]. The author co-created a project to prototype a mobile application that would facilitate the immediate answering of questions that fall within certain categories using a knowledge base, simple machine learning and profiling of users based on their usage history.

One of the initial goals of Shy was to bring knowledge via m-learning[2] as opposed to e-learning[3]. This is because m-learning systems are available to a much larger demographic of people, because the hardware (mobile devices, commonly phones) is becoming ever cheaper and more accessible.

A semi-functional prototype was completed for iOS, as seen in Figure 1. The front-end was complete, however in the backend question-answer matching was evaluated without knowledge of previous material the user had viewed. This meant that explicitly targeted answers for a question and suggestions of questions that a given user profile might be interested in could not be made. Up until this point, the service was restricted to working on iOS smart devices only, with poor quality question-answer matching. In the initial project, the knowledge base was built for questions on personal health, sex, relationships and family.

should add at least one more significant paragraph here, not sure what to write about though
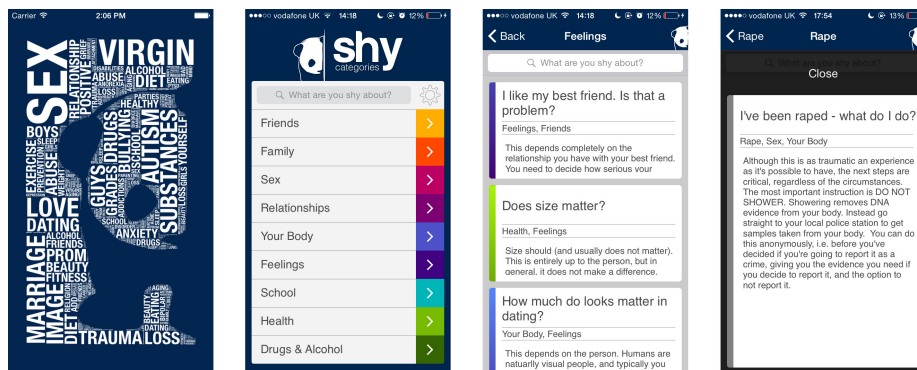
Figure 1: Screenshots of Shy iOS App

---

[1]http://toughhackers.com/yacht-hack-2014/)

[2]M-Learning: learning using tools available on a mobile device

[3]E-Learning: learning conducted online, usually with the help of a computer

2.2   Motivation for this project

Access to knowledge is a critical part of modern life. It's also a Human Right, under Article 27 of the Universal Declaration of Human Rights [1]. We have the facility to retrieve information and instantly communicate with each other using our smart phones, and commonly do it up to 200 times a day [2] - a facility that hundreds of millions of individuals live without. This project aims to investigate the technical and ethical challenges behind this and to create a technology to give more people access to knowledge, and thus their human rights.

The project involves the use of a number of systems to work including machine translation, an SMS input/output system and a system to build a profile on a user in order to facilitate high-quality targeted question-answer matching. The system will also feature machine learning so that as time progresses, the quality of the answers it returns increases. The use of these systems in one application along with complex ethics and privacy issues create an interesting project that draws together many technologies and discussions in a way that can be used as a basis for further research in the future.

2.3   Aims of this project

The first goal of this project is to assess and address the ethical issues that arise from the software that this project aims to create. These include the responsibility the software (and hence the software author) has, because of its position of trust, to provide accurate information and to protect the privacy of its users by storing only necessary information, among others. This project will do this by researching ethical issues relating to the technologies that the project will use, for example machine translation. This will then be used to feed the design process of the software and to specify the expected use-case of it. Finally, the resulting software will be evaluated through experimentation, with volunteers asking the software a set of questions within a topic area, in a non-English language, and evaluate the relevance of answers returned.

The initial goal for this dissertation was to focus on questions relating to personal health, sex, relationships and family, which was as a direct result of the focus of 'Shy'. Although a health and diseases database was located in the early stages of the project (Diseases Database, www.diseasesdatabase.com), it was targeted at professional uses for the data and came with a significant licence fee, as well as complex documentation that the provider customised specifically for each use-case. The author did attempt to acquire a free academic research licence via email but was unsuccessful. Additionally, as the project progressed in the early stages it became clear that the author's original two-page document discussing ethical considerations with the project did not cover all possible situations. Indeed, the Author and Supervisor were still discovering further significant ethical issues as far as eight weeks into the project, which would have required a complex ethics panel review, critically delaying this project. Two example ethical issues that occurred were:

objective enough?

- Any application has that dispenses health related information that a user might

act upon in a life changing manner has to respond with correct information, since an incorrect response could lead to a user taking an action which leads to them harming themselves.

- Privacy is critical since the questions been asked are personal. This could not be absolutely guaranteed, since a third-party service had to be used to send SMSs and, although unlikely, it was possible that this service could build profiles on users and identify them.

As a result of the above issues with locating a reputable knowledge base and the complex ethical issues, a decision was made in early December 2014 to switch the target topic for questions to Geography and to use Wikipedia as a datasource, since it is openly available, well populated and multiple methods exist for extracting information from it. In future work, it should be possible to extend this project to work with medical questions, if a high quality data-source is located and a solution to the ethical issues is found.

2.4   Structure of this report

This report starts with a review of pre-existing literature on this topic in chapter 3, where the author looks at existing software, tools and services of a similar type to those that will be used in this project. Comparisons between different software development life-cycles are also discussed.

Chapter 4 describes the method that will be taken to develop the software and service. This covers the software development life-cycle that will be followed, and sets out a plan for when development will take place. Requirements will also be identified, described and categorised as either functional or non-functional. A method of evaluating the success of the software will also be discussed and chosen.

The design of the software, driven from information collected from the requirements, will be set out in Chapter 5. The individual components that make up the technology will be described and discussed, followed by a description of how these modules will interface with each other to build a complete system. Finally, this section will include a description of the platform, language and other tools that will be used.

described and discussed, or just described?

Chapter 6 contains details about the implementation of the software. This includes information on the tools that were used to create components such as the SMS interface and question-answer matching system.

Results from the evaluation of the system will be presented in Chapter 7. A discussion of these results and their implication on this project is then presented to the reader in Chapter 8.

Chapter 9 contains an evaluation of the software produced and the process taken to complete it. Comparisons will be made to the success criteria, set out for the project in section ¡insert number here¿.

The author's thoughts on potential future work, building on the work produced in this project, are then displayed in chapter 10.

Finally, the main achievements of this project and then the report is concluded in chapter 11 with final thoughts.

2.5 Assumptions Within This Project

This project will make the assumption that users of this service have basic literacy skills in a language supported by the project. Although this is not a correct assumption world-wide, expanding the remit of this project to include a 'graphical' user interface that works over SMS would scale this project outside of the acceptable/expected range for an undergraduate dissertation.

The main use-case for this project is going to be parts of the world with limited internet access. Although these exist all over the world, the African continent is typical of this problem so I will focus my research in demonstrating a need for this tool there.

extend this

3 Literature Review

The point of this is to show that I 'own' the material and subject (competence). Show different points of view, choose a side, express a view on which I prefer and if I agree, critique. Have objective goals.

A significant amount of work has previously been undertaken in the area of making knowledge quickly accessible to us in the form of questions and answers. To take one example, consider that an individual needs to find the population of London as quickly as possible. The initial step would be to search for information. In the connected world, this is easy: a simple Google search returns the answers without even having to click through to any resources, as shown in Figure 2.

Figure 2: Google Now information box. Screenshot taken on 3rd February 2015.

This is easy to do in the developed world where up to 75% of the population are connected to the internet [3], with the population of elderly people been largely responsible for the remaining 25% [4]. This contrasts strongly with the situation in the African continent, where in 2013, internet usage had only reached 16% [3], a figure largely inflated by South Africa where 5% of the African population generate 2/3 of internet traffic from the African Continent [3].

3.1 Previous work using SMS

Although the above suggests that the African continent is majoritively disconnected, this is not the case. Because of restrictions in the electricity available, the ways in which a non-smart phone can be used in Africa have far surpassed those in the developed

15

cite that these are lower power consumption, or assume?
Emma: Isn't necessary but would be helpful; leave it up to whether you need to fill out your word

world [5], in part due to the low power consumption of a non-smart mobile phone. Interesting examples of SMS use in Africa include an automated service which sends an SMS to HIV/AIDS sufferers to remind them to take their medication, and a system for farmers that gives them the current market prices for goods, saving them from making regular long-distance trips to the market or relying on out-of-date information from a weekly radio broadcast [6].

Interactive systems have also been developed to operate over SMS. One successful example is mobile money platforms. These allow for users from any background to pay and be paid for goods and to transfer money across long distances at negligible cost [6]. One of the most highly adopted services is M-Pesa, which in Kenya alone was responsible for £5.7 Billion in transfers in 2012 [4]. M-Pesa gives users a balance linked to a national ID number from which they can pay for goods by sending an SMS with a cashier (recipient) number or pay outstanding bills in a similar way. Non-subscribers can also use the system by depositing money with a M-Pesa cashier in exchange for an access code, which can be sent via SMS to a contact who can subsequently redeem it with their local cashier [6].

This difference in standard use of mobile phones is demonstrated in the International Telecommunications Union's 2013 report [3], which shows that in Europe, for 790 million mobile subscriptions, 53% of subscribers have mobile internet access (422 million), compared to 17% in Africa (93 million have mobile broadband, out of 545 mobile subscribers). This is due to the prohibitively high cost of accessing data services, regardless of the hardware that the user has. In 2012 in Europe, 500 MB of data per month for 12 months cost 1.2% of the average Gross National Income Per Capita (GNI pc). In Africa, the average price was 30x this, at 36.2% of an individuals GNI pc [7].

3.2 Ethical Issues

This project raises a number of ethical issues related to translation accuracy, providing information to people in an ethically acceptable way (with focus on information accuracy) and maintaining user privacy.

3.2.1 Ethics of Providing Information

One significant issue for this project stems from providing information that may affect an individual or lead them to take a harmful action.

In a similar way to that which a teacher has a responsibility to teach accurate information to a pupil due to their position of trust, any service relied upon by a user must equally provide accurate information. The result of not doing this

---

[4] Data from Safaricom, M-Pesa operator in Kenya - actual value 817,085,000,000 Kenyan Shillings, converted to GBP on 1st November 2014 at rate of approximately 0.007.

could be providing inaccurate information that leads to a user carrying out an action that causes harm.

this needs to be waaaayyyyy expanded, but I struggled to find information that was useful. There's lots on remote education, and on basic ethics of teaching, but I struggled to find anything specifically relating to ethics of ensuring information you provide is correct.

3.2.2 Ethics of Translation

A significant part of this project is represented by the support of multiple languages. It is clear that translation on demand, at scale, needs to be automated by some kind of machine or algorithmic translation.

This project involves two blocks of translation. These are:

- The translation of the user's input into the language of the system (English)

- The translation of the answer to a user's question from the system language to their local language.

The second of these raises some issues, but to understand these it is first necessary to understand the two categories of machine translation.

Rule-based machine translation effectively treats human language in a similar style to programming languages. Formal grammars and lexicons are used to represent words that exist in either the source or target language, structures representing the translation of individual words or groups of words. Map structures contain mappings between individual or groups of words to their translated counterparts, sometimes with multiple results (a 1-to-many map), from which rules decide which is selected. Maps and rule sets are created by trained computational linguists [8].

More commonly, Statistical Machine Translation (SMT) is used, for example, by Google and Microsoft in their translation products [8, 9]. SMT learns mappings between strings of words of potentially unequal lengths from pre-existing original texts and their trusted human translations. The accuracy and breadth of language support for translation increases as more source material is analysed by the system, as potentially erroneous or low quality translations can be identified and flagged. SMT is also dependent on the quality of the human translation on the input material [8].

A significant issue that is present in statistical machine learning systems comes from the knowledge we assume an individual has and derives from a word; in this project, this affects the translation of the answers returned to a user. In human translation this is solved by the translator's knowledge of the difference in material culture, allowing them to append necessary information to the resulting translation that the recipient might find useful. In statistical machine learning, cultural awareness of material knowledge is a separate problem on its own. **formal reference to this, probably**

again from Kenny, but should be able to find better. To take one example from Melby (2006):

> "when translating a French menu, a human translator might stop to think that an English speaker in France would appreciate being told that a steak tartare is served entirely raw, even if this information is not contained in the original text (because French people might be assumed to know this already). Such a translator would be aware of differences in material culture, and would be able to empathise with the English speaker who might choose to avoid the dish, given more information" [8, 10]

This issue is a result of the translation engine not being capable of take and use a complete representation of the expected cultural differences between those who speak the input language and those who speak the output language [10].

below should really be expanded and be in the discussion. Explain the decision to keep translated answers in the database, which are pre-approved, to ensure no cultural misunderstandings.

The potential uses of this system include distributing information that may be used by individuals making important decisions, regardless of whether that is within this project or in further work based off of this project, as cultural confusion such as this could have potentially catastrophic effects.

Lilian: quote taken from [8], but quote is interpreted from [10] - so I think referencing both is correct, but for this reason rather than last reason we discussed?

3.2.3 User Privacy and Data Protection

This project uses a simple Artificial Intelligence (AI) to keep a record of information that has already been returned to a user, to attempt to save sending repetitive information (saving the cost of additional SMS). This raises another ethical issue though of privacy. The material that a user has researched is likely to be sensitive; such is the nature of health related questions. This means that the information on a user has to be kept securely, in a way that an individual user can not be identified.

One way of doing this is using a technology called hashing. Hashing is the technique of taking data as an input and generating an output value (called a 'hash') that is unique to an input. Hash functions are ideally one way functions where given a piece of input data X, the output hash will always be a constant value Y, whenever and however the hashing algorithm is executed. This allows a user's identity (phone number) to be obscured, in such a way that the mobile number can not be recovered from the database. When a question is received, the phone number (represented by variable X in the above example) can be hashed and the data for this number looked up from the database without the database application ever being aware of the phone number of the user. In the example in Figure 3, a phone number (X) is used as input to an imaginary hashing function and the output hash (Y) is shown on the right hand side. As just described, this number can then be used as the key for the user information database, shown in Table 1 to retrieve the information needed for the AI.

userId	userData
8f64B2	{previousQuestions:[13,301,170,577]}
9ac5e3	{previousQuestions:[441,56]}
f9dd7e	{previousQuestions:[301,623,89,280,364,621,209]}

Table 1: Example of software database: hashed phone numbers paired with a list of previous returned answer ID's.

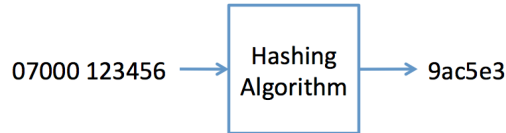**reference the above from security engineering ross anderson, chapter 10 and 21?**

Figure 3: Hashing a phone number using an arbitrary hashing algorithm

In practise it is possible for the output of a hashing algorithm not to be unique to a particular input. This is known as the collision resistance property of a hashing algorithm and formally is the property of a hashing algorithm that determines the probability of the same output hash for two distinct random inputs [11]. It occurs because the range of possible input values is much larger than the range of output hashes, and because all valid inputs must map to an output there is repetition in outputs [11]. This property of hashing algorithms can be a security risk for some use-cases, where hashes are used to verify that data has not changed since been hashed. In this case, a third-party may be attempting to generate alternative, malicious data with the same hash as some pre-existing data, to swap them undetected [12]. In the use-case of this project there is no security risk since hashes are only used to anonymise phone numbers, however a high collision resistance is necessary to ensure that two users of the system with distinct phone numbers don't get mapped to the same row in the database, resulting in poor quality question-answer matching (or in the extreme worst case, privacy issues originating from returning to one user answers that are targeted to another).

Irreversibility is a necessary property of the hashing algorithm chosen for this project. This ensures that the key used in the database cannot be used to retrieve the phone number of the user, thus protecting their identity. Commonly used hashing algorithm families, such as the MD family (e.g. MD5) and SHA family are all designed to be one-way [13][5].

---

[5]Bruce Schneier is a fellow at Harvard's Berkman center for internet and society. He has been posting regularly for his newsletter and then blog since 1998 (over 16 years) and has published material related to this field throughout this time. He has been involved in the creation of other cryptographic algorithms, such as the Skein hashing algorithm, blowfish block cipher, so may have a vested interest in dis-crediting other

Although hashing algorithms are computationally one-way, they can suffer from another type of vulnerability affecting the security of the original input data. When hashes are used for short strings of information (passwords, for example), a simple way to try and retrieve the original data is to compute a dictionary of the hashes of lots of common passwords, from which matches can be found. This idea has been extended to produce Precomputed Hash Chains and subsequently Rainbow Tables. These are highly efficient data structures that consist of chains of processed input messages and their hashes (with a reduce function to reach the next item in the chain), allowing original input data to be looked-up from its hash. Both Precomputed Hash Chains and Rainbow Tables tend to be Terabytes in size, and as such they only exist in a significant form for the most commonly used hash functions (including MD5 and SHA) [14]. Within this project, this raises privacy implications, as hashes within the database would be 'convertible' to phone numbers.

perhaps the below should move to method and requirements - TODO needs to be cleaned up as well.

In password authentication systems, this can be solved by the use of a technique called Salting [15]. Salting is where a 'salt' (a piece of additional unique data) is added to each piece of input data before hashing, and then this salt is prepended to the output hash. This nullifies the use of Precomputed Hash Chains and subsequently Rainbow Tables [15]. When checking if a plaintext password is equal to the hashed and salted representation, the salt from the stored hash is added to the plaintext password, the hashing algorithm applied and then the output compared to the stored obscured password. However, in this application this would prevent efficient lookup of users from phone numbers in the application, as the application would have to compare an input phone number against every user object in the database, which is computationally expensive and slow since a salted hash has to be computed for potentially every user record [15].

As such, it has been decided that the best way of protecting a user's phone number in the event of a database hack without significantly degrading performance is to salt phone numbers with a constant, large and complex salt (meaning a rainbow table would have to be specifically computed specifically for this application and would be extremely large in size).

3.3 Software Design Life Cycles

There are many software development life cycles that a developer or company has available to choose from. Four common ones include the Waterfall Model [16], Itera-

---

algorithms such as MD5 and SHA. However, the essay cited above is written as a result of the Crypto 2004 Conference in California, Schneier has a world-renowned reputation for cryptography and computer security and the algorithm that he suggests should be used at the end is not one that he has involvement in. For these reasons, I believe this source is reliable.

tive Model [17][6], Spiral Model [18] and the Agile Development Model [19][7]. In order to choose which life cycle suits this project best, research into all four has been carried out.

3.3.1 The Waterfall Model

The waterfall model consists of 5 phases of work, where each leads directly into the next. It is a plan driven development model, which means that all process activities such as the implementation of the project must be planned and scheduled before work on them begins [20]. The five layers are shown in Figure 4 and described below:

1. **Requirements**. In this stage, both user and system requirements for the project are determined after examining the business goals for the project. These form the system specification.

2. **Design**. Here, a plan for the software development is created. The overall architecture for the software is created, the external libraries to be used are chosen and the interaction between different modules is modelled.

3. **Implementation**. In this stage, the software is implemented to match the design and architecture set out in the Design phase. Unit tests are written to ensure that each module matches its specification.

4. **Testing**. In this section the system is tested as a whole to check for any bugs that might not be picked up when testing individual modules in isolation. The software is then delivered.

5. **Maintenance**. Finally, the software is deployed and maintained. This includes making changes to the requirements and subsequently the software (implementation) as the business goals change. [20]

A key disadvantage of the waterfall model is that once development has moved from one stage to another it cannot go back because the model is fixed and progresses in one direction only, until the maintenance step. The main advantage of the Waterfall Model is its simplicity: it is easy to understand and to use, saving time at the start of the project that would otherwise have been spent on learning and preparing to use a more complex model.

---

[6]Iterative development came together from many concepts dating back to 1957. There was no single paper that initially presented the entire concept, but this paper gives a summary of how the various concepts came together into the current Iterative Development model.

[7]The Agile Manifesto brought together a group of Agile Methods which had been in use since the mid-1990's into a single concept - that of Agile Development.
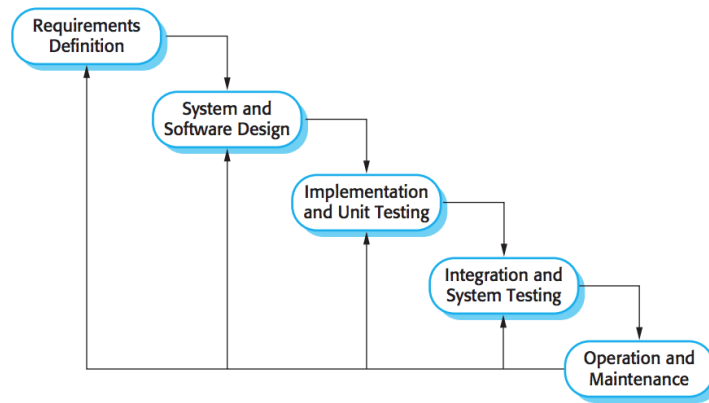
Figure 4: Waterfall Development Model
source: Sommerville [20]

3.3.2 The Iterative Model

Iterative development is the practise of splitting the overall development of a project into multiple independent and distinct blocks. A block contains a sequence of tasks (essentially a mini life cycle) such that each iteration can really be thought of as its own project. Each iteration tends to focus on a specific set of features such that when the iteration is complete the overall project is stable and the modules developed to-date can be tested as a whole system [21].

There are a multiple variations of the iterative development method, including Timeboxed Iterative Development, where all iterations have a fixed inflexible duration, and Client-driven Iterative Development, where each iteration contains the features that the client considers have the highest business value [22].

A common variation is based on the waterfall model and is shown in Figure 5. The first three tasks in the cycle represent the first four stages in the waterfall model (Requirements, Design, Implementation and Testing), at which point there is the option to either evaluate progress so far, and start another iteration, or to deploy the software and end development. One significant difference between this and the waterfall model is that maintenance is not represented.

A key advantage of the Iterative Model is that it allows for the user to revise their requirements as they progress, making it dynamic and flexible. This is also its disadvantage, in that the lack of fixed structure can be a weakness unless the project is planned well.
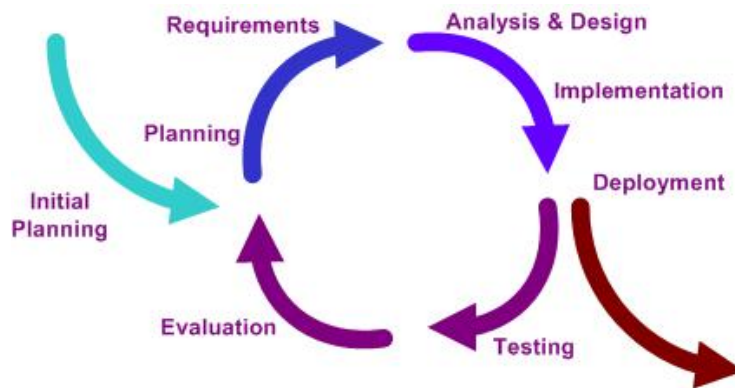
Figure 5: Iterative Development Method
source: `https://commons.wikimedia.org/wiki/File:`
`Iterative_development_model_V2.jpg`

3.3.3 Boehm's Spiral Model

The Spiral Model was originally developed as a result of the adjustments that were commonly made to the waterfall model in large government projects [18]. The Spiral Model is a general model and can be used as a generator for other more specific models, given the parameters for a project such as its risks [23]. Models such as the waterfall model are specialisations of the spiral model [23].

Each loop represents a stage of the software development, from planning on the inside loop to system testing and verification on the outside, as shown in Figure 6.

A key advantage of the Spiral Model is the level of detail it models. The stage of a sub-task within the entire development process is represented by the distance from the origin, and each loop is split into small, well described sections. A key disadvantage is that the planning and use of the model is largely driven by the results of complex risk analysis on the sub-tasks [20]. Although this makes it favourable to large organisations, it has the opposite effect for individuals.

A detailed description on the spiral model can be found in Boehm's paper "A spiral model of software development and enhancement", where he published the complete concept [18].

3.3.4 Agile Development

Agile Development essentially represents a group of development techniques targeted towards either the development of a small to medium sized product for sale or the development of a custom system within an organisation, where the customer has a high degree of involvement (for example 'Extreme Programming') [20].

23
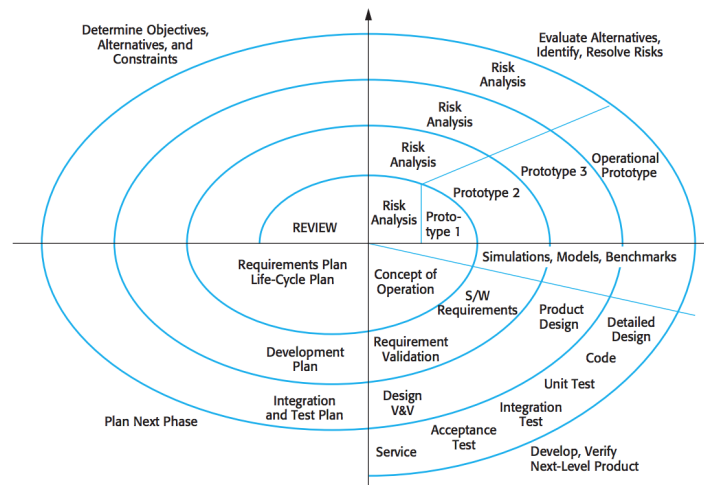
Figure 6: Boehm's Spiral Model
source: Sommerville [20]

Agile development is highly targeted towards teams, with components such as Pair Programming, Collective Ownership, regular meetings and employee health (avoiding overtime) playing an integral part in the model [20].

3.3.5 Development Life Cycle Summary

Iterative development has been chosen as the model that this project will follow, based on the following factors:

1. Boehm's Spiral Model and Agile Development are both targeted towards development teams. A significant part of these models relates to team interaction, so they are not appropriate to this project.

2. The Waterfall Model is rigid and inflexible. It does not allow for the changes to the requirements and implementation that might be proposed after evaluating the quality of the first implementation.

3. The Iterative development model can be implemented effectively by a single developer , and supports the adjustment of requirements and implementation in each iteration of the cycle.

I need to add this as a 'pro' above

24

4   Method and Requirements

4.1   Plan for Software Development

The Iterative Development Model was selected in Section 3.3.5. Therefore, the development of this service will follow the principles set forth in this process.

4.2   Requirements

4.2.1   Use Cases / User Goals

Two personas were created in Appendix Section 12 to represent individuals who may have a need for the work resulting from this project. These are Nanjala, who is from Uganda, and Cedric, who is from Switzerland, and it is from these that user goals are extracted. These describe what a general user aims to achieve with this service.

1. Nanjala wants to be able to read general information on any given topic.

2. Cedric wants to be able to retrieve facts from direct questions.

3. Both Nanjala and Cedric want to be able to achieve their goals entirely via SMS because of their circumstances.

4. Cedric needs a simple and efficient user interface that allows him to retrieve facts quickly.

5. Cedric wants to be able to feedback into the system when he finds an answer particularly useful or unhelpful.

6. Nanjala wants to be able to show the service to her friends without explaining how to use it every time.

7. Nanjala wants to get recommendations on content she might find interesting.

8. Cedric wants the system to protect his privacy, obscuring his identity.

Note to self: might not implement this, in which case discuss in DISCUSSION.

4.2.2   User Requirements

User goals are distilled into formal user requirements. Each user requirement corresponds directly to the correspondingly numbered user goal.

1. Users must be able to interact with the system using natural language.

2. Users must be able to request general or descriptive information on a geographical entity.

3. Users must be able to ask questions requesting a fact or statistic about an entity and get the answer or number they are looking for.

4. Users must be able to interact with the system over SMS.

5. Users must be able to make their queries and get responses quickly.

6. Users must be able to feedback to the system when an answer is of noticeably low or high quality so the system can learn and improve.

7. Users must be able to learn how to use the system themselves and this learning process should be quick.

8. Users must be able to get recommendations on entities they might be interested in, based on previous query history.

9. Users must be comfortable that their privacy is been protected whilst using the service.

Haven't decided yet if this is a feature I will implement - time based!

4.2.3 System Requirements - Functional

1. The system will be capable of processing queries in natural language form.

2. The system will receive queries via SMS, process them and then respond via SMS.

3. The system shall use a datasource that contains both facts and descriptive text on geographical entities to answer queries.

4. The system must be able to parse queries to identify the entity that is been asked about and the specific parameter of that entity that should be returned as the answer to the user's query.

5. As well as giving direct facts and statistics, the system should be able to give general descriptive information on an entity.

6. The system must be able to match words of similar meaning to the parameters in the data source. For example, the word 'long' must be matched to the parameter 'length'.

7. The system must learn from user feedback when answers that are returned are of high or low quality an apply this knowledge to future queries.

8. The system must be self-explanatory, with the facility to offer instructions to a user.

9. The system must use user state so that it can accept queries that are dependent on past questions (for example, a query asking for more information or feedback on the quality of a past question).

Non-Functional Requirement	Fit Criterion
The system behind the SMS gateway must respond in a timely manner.	From an SMS been received, a response must be sent within 20 seconds. SMS transmission times on the cellular network and gateway are out of the control of this project, so are not accounted for.
The answers returned must be of high quality.	In testing, the system returns the correct information for at least 90% of queries initially (before learning starts) if an answer is returned at-all.
Answers that are consistently rated poorly are are less likely to be returned.	In testing, after receiving a small number of negative ratings for an answer, a different answer is returned.
Instructions to use the service should be clear and concise, with no ambiguity.	In testing, having explained the purpose of the system, a user should be able to use the system with just the help information, without needing to ask for assistance.
The system should accept and be able to process questions in natural human language as opposed to a delimited message.	In testing, a user should be able to use the system for general questions without having had a syntax explained to them.

Table 2: List of non-functional requirements and their Fit Criteria.

10. The system must be easy to implement and flexible, to accommodate future projects that make use of adaptations of this project.

11. The system must obscure a user's phone number in the database.

4.2.4 System Requirements - Non-Functional

Non-functional requirements are listed in table 2 and are accompanied by a Fit Criterion, which will be used to identify whether a requirement has been met successfully.

4.3 Data Sources

A number of possible datasources were identified from which data could be drawn from for this project. These were all based off of Wikipedia (a large, linked source of knowledge), and include the Wikipedia API and various representations of the DBPedia Ontology. In this section, both will be discussed and a decision will be made with regard to which ones are used.

4.3.1 Wikipedia API

The Wikipedia API is a standard MediaWiki API and allows applications to query a page or resource for various parameters from a page, including the page contents, infobox parameters, citation list and links to other articles in various flat or multi-layered data structures [24]. Interactions are made via a HTTP GET request to a URL containing the parameters, and responses can be received in JSON. This makes the API easy to use and widely compatible, especially since JSON can be converted to a dictionary in most languages.

4.3.2 DBPedia Ontology

The DBPedia Ontology is the result of a project that aims to build an OWL ontology representing all of the knowledge within Wikipedia. The ontology has been manually created from the most commonly used infoboxes on Wikipedia and covers 685 classes with 2,795 properties, providing structured information on 4.2 million places, people, work, species and organisations [25]. This makes it especially good for use in applications since the data is both curated and well structured.

There are two ways of querying the DBPedia Ontology. One is using the standard SPARQL ontology query software to make a query form the ontology structure, however this is complex as queries must be structured correctly and it is necessary to know the class structure of the entities been queried [26].

In the event that the class structure is not known, queries can be made by querying the top-level representation of an entity in the ontology, which presents data in a similar format to that of the Wikipedia API, albeit in a 'cleaner' form (with human readable keys) as a result of the data been processed by hand. Although the keys are of a more human-readable format, some material is abstracted within the ontology, and so is inaccessible with a basic query. An example of this is page for 'London', which does not have a parameter 'Mayor' with the value 'Boris Johnson'. Instead it refers to a number of 'Leader Titles' and 'Leader Names', which include a reference to Boris Johnson[8].

4.3.3 Choosing the Source and Interface for this Project

This project has a limited scope and time period available to it, and creating a system capable of identifying the relevant classes and then constructing SPARQL queries for looking up an abstract property is an example of an approach that is out of scope (as it involves natural language processing and semantic analysis of the names of the ontological classes - not just the query been made). As such, this project will make use of a combination of the data from the Wikipedia API and DBPedia top-level entity representations, collecting keys from both and working with the key with the highest semantic relatedness match to identify the data-value that most likely is the answer to a

---

[8]A web representation of the London object is available at: http://dbpedia.org/page/London

user's question. The DBPedia Ontology also provides access to the abstract at the same level as the infobox, so general descriptions of entities will be sourced from there.

4.4   Processing Natural Language Queries

Because of the nature of the audience for the system been developed in this project, it needs to be able to process input in natural language form rather than enforcing a formal grammar on users (as described in User Requirement 1 and System Requirement1). This involves constructing a natural language processing system to process queries and identify their meaning. In this project, queries are of a simple form - if a user is asking a question they are requesting a property of information about a given entity. As assumption is therefore made that each query consists of: a token representing the geographical entity been queried; a token representing the property been requested of that geographical entity; and some noise in the form of stop words [9].

It necessary to be able to identify the geographical entity in the query. This will be done using a Named Entity Extraction tool (of which there many tools available online, including Cortical[10] and Dandelion[11]). These tools extract keywords from a text and in some cases are capable of categorising keywords according to an ontology. For example, Dandelion links entities to the DBPedia Ontology [28].

The property about the geographical entity will be established by removing stop words, punctuation and the name of the geographical entity from the string. In the input 'What is the height of the Eiffel Tower?', this would result in the string 'height'.

The result of this simple natural language processing should be that a sentence such as 'What is the Height of the Eiffel Tower?' is reduced to a structure indicating that the geographical entity is the 'Eiffel Tower' and the property is the 'height'.

4.5   Evaluating Success

Success will be evaluated by establishing if the project fulfils it's original requirements, based on the original scenarios. Many user requirements such as protecting the privacy of a user will be proven through software, whilst some that involve user interactions must be tested with users. As such, user testing will be used to evaluate three specific properties of the software through a trial usage and subsequent questionnaire. The questionnaire will aim to evaluate the usefulness of the system, the usability of the system and the quality of returned answers, and is based off of the IBM "Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology" questionnaire [29]. Questions were modified to change to suit the software been used

---

[9]Stop words are words that are frequently used in in language but do not carry any significance themselves [27]. It is common practise in Natural Language Processing to to remove these stop words before attempting to process an input [27].

[10]http://www.cortical.io/demos/keyword-extraction/

[11]https://dandelion.eu/products/datatxt/nex/demo

in a situation without internet as opposed to a general workplace environment, and a question was added to investigate the quality of answers returned to a user. A copy of the questionnaire to be used is included in Appendix 12.2.2.

5 Design

In this section, the design of the system will be described, including information on: the structure of the server application and database; interactions with third-party services; systems used to protect user-identity; and the language, platform, tools and libraries used for development.

5.1 System Design

In this section, the high-level design of the system software and its interactions with third-party services is described. The system design was driven by requirements listed in sections 4.2.2, 4.2.3 and 4.2.4, which were generalised to:

- The system must be modular with points to add additional plugins for example, for additional data sources.

- The system must be easy to maintain, with distinct boundaries between functional components and behaviour.

- The database must be local to decrease the risk of exposing data.

- The system must have clear interfaces for interacting with the external world.

Based majoritively from these requirements, the formal aforementioned requirements and the application of an informal gap-analysis between each iteration of the architecture, the system architecture shown in Figure 7 was created. This shows the structure of the system, from the level of hardware and third-party services through to individual software modules.

5.1.1 Micro-system for Protecting User Identity

In section 3.2.2, the need to protect the privacy of a user and the general technique that will be used (involving irreversible hashing) are described. The design of this system will now be discussed.

The main gateway into the system will be an HTTP call to a Python function, which takes the phone number of the user and the question being asked as its parameters. This function represents a core part of the privacy protection system, by generating the hashed phone number and only using that when calling functions that compute an answer. The user's phone number is only subsequently used when

```python
def entry(userCell, question):
    hashedCell = someHashFunction(userCell)
    answer = findAnswer(hashedCell, question)
    return userCell, answer
```

31
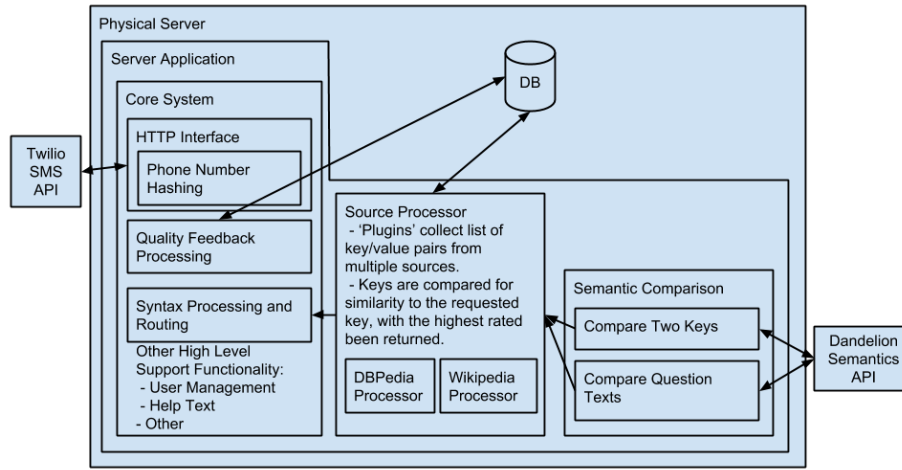
Figure 7: System Architecture

The userCell is required at the output of the function to determine the number to which a response will be sent. However, as the computation of an answer only requires a general constant representation of user identity, not necessarily the user's phone number, the specific value of userCell is not required within the computation of the answer. As such, a hash is passed into the answer-computing function. This hash will be used within the database to store a user's history and profile. This means that only a user's history can be identified from the database, not the actual identity of a user.

5.2 Language, Platform and Tools and Libraries

5.2.1 Language

The language Python has been selected to be the main language in project for four reasons:

1. Python is universally compatible and simple to execute. Once the server application is written, it will be simple to migrate it between machines and adjust the application's configuration or environment.

2. Python is commonly used and has significant community support in the form of libraries for all the tools that will be used in this project (for example, libraries for querying common databases and for sending text messages). This is useful as it will prevent the author spending time implementing common functionality like database calls.

3. Python is an extremely flexible language in which it is easy to implement applications of this type without syntax issues. This fulfils the system requirement in section (4.2.3, point 10) that the application is easy and quick to implement.

4. The author has significant expertise in the language, as does Lilian Blot (dissertation Supervisor).

A testing utility will also be created to query the application from a personal computer without using SMS. A simple OS X Cocoa application will be created that queries the HTTP interface with JSON in the expected format and displays an answer. Objective-C and and Mac OS X were chosen as the language and platform for this as the author has experience with Objective-C and uses a Mac. There was no requirement for universal compatibility because the application is only for testing and development purposes.

5.2.2 Platform

The development platform of the utility will be Linux. This is for three reasons:

1. Linux manages Python and its libraries in a simple way without complex installers with tools like the Python Package Index (a library repository) [30].

2. Common libraries that will be used for creating a HTTP server, creating a REST API, making database queries and sending SMSs tend to be written primarily on the Linux platform.

Lilian: should I explain what this is somewhere?

3. The author has familiarity with Linux and has access to significant Linux infrastructure under a grant from Rackspace.

As previously stated, the interface to the application will be over the SMS platform.

5.2.3 Database

There are three tables of data that need to be stored. These are the tables for user profiles (to allow the application to handle queries that make use of 'state' with users), for the machine learning of question-answer quality, and for an initial intelligence that a new deployment of the application starts with. Both the conceptual data model and the logical database design are now presented.

Conceptual Data Model
There are five entities in the conceptual data model for this project. These are:

- **Users:** An entity that represents a user, purely by an identifier based on their phone number.

- **Complete User History:** A history of all questions they have asked previously to allow for future expansion of the system, perhaps to implement a recommendations system.

- **Last Question Asked by a User:** An object containing the parameters of the last query. This allows the system to handle stateful queries based upon a previous query.

- **Keyword Pairs Quality Learning:** Will contain the property requested by a user, the property returned to that user, and the number of times a matching was ranked as been either one, two, three, four or five stars.

- **Initial Intelligence:** Undecided.

These entities shown in Figure 8, which also shows the entity relationships.
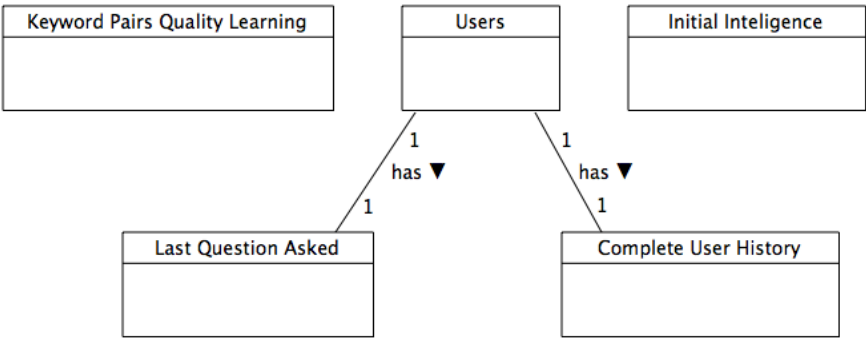
Figure 8: Conceptual Model of Data

Logical Database Design
The logical design of the database reduces the number of entities from five in the conceptual model to three. This is because there is a one-to-one relationship between the *User* entity and both the *Complete User History* and *Last Question Asked by a User* entities, with a small number of fields. In a large system where relational databases were used it would make sense to keep these entities separate, but in this case merging them allows a wider variety of databases to be used, simplifies the implementation and does not add significantly add to the complexity of the data model.

The logical database requirements are listed in Table 3 and represented graphically in Figure 9.

Database Technology
The database technology requirements for this project are:

Need to fill in initial intelligence

Table/Collection	Parameters
Users	(<u>id</u>, cellNumber, history, lastQuestionText, lastQuestionRequestedProperty, lastQuestionReturnedProperty)
KeywordRelatednessQuality	(<u>id</u>, givenProperty, returnedProperty, oneStarRatings, twoStarRatings, threeStarRatings, fourStarRatings, fiveStarRatings)
InitialIntelligence	Undecided.

Table 3: Logical Database Design. <u>Underlined</u> fields represent primary keys.
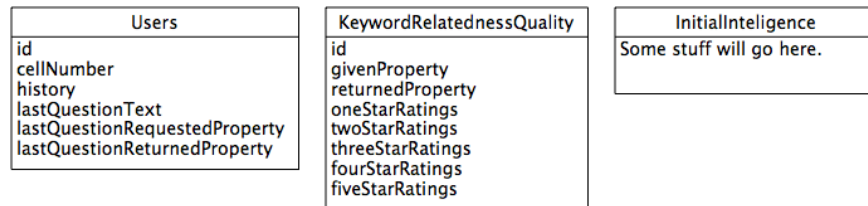
Figure 9: Logical Model of Data

1. The selected database must be as simple in implementation and interface as possible, whilst still supporting the system requirements. This is to allow for quick development and to avoid adding unnecessary complexity.

2. The selected database must allow for storage and retrieval of records with multiple parameters.

3. Support for records that are related to each other is not required; so relational databases are not required, but not ruled out.

4. The database needs to have built-in support for Sharding and Replication/Mirroring to allow the application to scale.

5. The database must have a library for interfacing with it in the language of this project.

There are a number of database technologies available that would fulfil the data requirements of this project. These can be split into the categories of relational databases (SQL) and document stores (NoSQL). Two common database approaches that meet the above requirements are MySQL with an ORM [12], and MongoDB with the Mongo Client. Table 4 shows a comparison of both of these technologies.

---

[12]ORM: Object-relational mapping tool, which maps queries from the syntax of your programming language to SQL. ORM's commonly also apply optimisations to queries, and validate them against SQL injec-

	MySQL/SQLAlchemy	MongoDB
Fixed Schema	Yes, reducing flexible	No, giving flexibility
Querying Simplicity	Requires complex ORM and awareness of transactions, or construction of SQL queries.	MongoDB comes with a Python client, which allows queries in the form of dictionaries.
Relational Queries Supported	Yes (could be of help in a future extension)	No
Records support nested parameters	No	Yes
Built in Sharding support	No	Yes
Built in Mirroring support	No	Yes

Table 4: MySQL & SQLAlchemy vs MongoDB

From this comparison, the decision to use MongoDB has been taken because MongoDB has more flexibility than SQL, writing queries is easier in Python and, other than the possibility of future extensions requiring relational structure, it is more scalable than SQL.

Physical Database Design
MongoDB separates entities into collections but does not place restrictions on the structure of objects within a collection. The general structure of the keys of a document (a JSON dictionary) is shown in Table 5.

5.2.4 Libraries

Database Querying
MongoDB comes with a Python client called PyMongo[13], which is developed by the same developers as MongoDB. It is the recommended tool to interact with a MongoDB database [31] and has good community support, so it will be used in this project.

Creating HTTP Server and REST API
The application needs to have an HTTP interface to allow interactions with it via SMS. As a result of the non-functional requirement "The system behind the SMS gateway must respond in a timely manner" in section 4.2.4, the application will need to be able to process multiple HTTP requests concurrently to prevent requests queueing up and responses been significantly delayed.

To create an HTTP Interface, the latest Web Server Gateway Interface (WSGI) specification (version 1.0.1, published as PEP 3333) [32] will be followed as it is the

---

tions.

[13]For more information: https://api.mongodb.org/python/current/

Users	```
	{
	'_id',
	'cellNumber',
	'history',
	'lastQuestion' : {
	    'text',
	    'givenProperty',
	    'returnedProperty'
	    }
	}
	``

KeywordRelatednessQuality	```
	{
	'_id',
	'givenProperty',
	'returnedProperty',
	'oneStarRatings',
	'twoStarRatings',
	'threeStarRatings',
	'fourStarRatings',
	'fiveStarRatings'
	}
	``

InitialInteligence	Something here Undecided.

Table 5: Typical Document in each MongoDB Collection

industry standard . The WSGI specification describes a general interface between web servers and web applications or frameworks (such as Twilio) in the Python language, thus describing how the web server should be constructed.

In this project, two tools are needed: a WSGI server, and a framework for writing WSGI applications. Flask will be used for the framework as the author has extensive experience using it and it is 100% compliant with the WSGI standard [33][14].

With respect to the WSGI server, many implementations exist such as a simple WSGI server included in Flask [33] called Werkzeug [34], and the Gevent WSGI server [35]. Support for multiple concurrent HTTP connections is not a requirement in the WSGI specification, and so some WSGI servers such as the implementation of Werkzeug in Flask do not support this[15]. Gevent does support concurrent HTTP connections [36][16], and so will be used in this project.

Sending SMS
There are a number of SMS providers with HTTP gateways available to choose from, all of which are similar. For this project, the provider must match the following criteria:

- Forwards received messages to a REST API.

- Supports sending messages via a REST API.

- The provider must have good coverage of countries so that users from many countries can use the service.

- The provider must have a 'good' reputation. User privacy is still an issue, since data is unencrypted when been sent over SMS, so the provider should be trustworthy. This is difficult to measure accurately, but one possible way of doing this is by looking at the number of significant clients a provider has.

Two services considered were that matched these criteria. These were Twilio[17] and Plivo[18]. Both use REST API's for handling incoming messages and for sending responses. Both Twilio and Plivo have comparable coverage as well - Twilio state that they send and receive SMS with numbers in 198 countries [38], and Plivo state that

---

[14]100% compliant according to the Flask product website.

[15]Werkzeug does support processing multiple HTTP requests concurrently by specifically enabling threading and increasing the default maximum number of processes from one (as described in the documentation at: http://werkzeug.pocoo.org/docs/0.10/serving/ [accessed: 4th April 2015]). However, the implementation of Werkzeug in Flask has these disabled in the source code because Flask uses thread-local storage (memory that is local to a single thread) and so can not support concurrent HTTP requests on new threads. This is stated on the Flask website at http://flask.pocoo.org/docs/0.10/design/#thread-locals [accessed: 4th April 2015].

[16]Gevent using the Greenlet library [37], which provides a notion of micro-threading (coroutines). When a new HTTP request is received, Gevent creates a new Greenlet, thus preventing the main thread been blocked and allowing multiple concurrent HTTP requests. The creation of the Greenlet is described in the preceeding reference([36]).

[17]Twilio website: https://www.twilio.com/

[18]Plivo website: https://www.plivo.com/

they support 207 countries [39][19]. The difference is trivial, as the number of recognised independent states in the world is between 190 and 200 (depending on listing) - the US Government recognises 195 [40].

With regard to reputation, for this project it was decided that Twilio is preferable to Plivo. Although it is difficult to measure reputation accurately, Twilio was founded in 2007 and has an group of clients including PayPal, Uber, Sprint and Airbnb, compared to Plivo, which was founded in 2011 and mainly works with entertainment companies.

Semantic Analysis - Measuring semantic relatedness between texts
Semantic comparison of text will be used for two purposes within the project: to match keywords from datasources (for example, if a user asks `how tall` something is, the property `height` would be appropriate to return; and to match questions with identical meaning to each other to save the application doing a full analysis of the data sources again. The requirements for these use-cases are different: for the former the tools needs to compare pairs of either single words or short strings or words for semantic similiarity; and for the latter the tool needs to be able to compare full sentences for semantic similarity.

Semantic similarity is a measure of how semantically alike entities are in their meaning (for example, `bank` and `trust company`) [41]. A more general measure exists, called Semantic Relatedness. Whereas semantic similarity compares entities by their meaning, entities may be semantically related by additional lexical relationships such as antonymy (the opposite of an entity, for example, `tall` and `short`) and meronymy (entities that are a part of another entity, for example, `car` and `wheel`) [41, 42]. Both use-cases in this application will be using semantic similarity as semantic relatedness is too general. An example using the datasource keyword matching use-case is that using semantic relatedness, the word `tall` would be related to `width` as well as the target word of `height` as a result of antonymy, potentially cluttering results. In the second use-case, consider a highly rated question in the database of `What is the population of England?` and then the question of `What is the population of London?` been processed. London is a part of England, so there is the potential for the population of England to be returned instead of the population of London.

A key requirement for the algorithm and method used to compute semantic similarity is that it can be done so in compliance with the system requirement in section (4.2.3, point 10) - that the system is easy to implement. A number of algorithms were discovered for calculating semantic similarity based on the WordNet lexical database [43], such as the LeacockChodorow [44] and Resnik [45] algorithms. Although mathematical representations of the algorithms are published, to fulfil the aforementioned requirement a pre-implemented semantic comparison tool is needed as implementation of these algorithms would be a non-trivial software engineering project. Both of the LeacockChodorow and Resnik algorithms are available in a Perl package as part of

should I move definitions to the footnotes?

use quotation marks in the above?

---

[19]This is referenced on a marketing page (https://www.plivo.com/twilio-alternative/) where Plivo compare their product to Twilio's, so is likely to be biased

WordNet::Similarity, a set of Perl modules with an optional web interface for integration into applications. No other open-source libraries were found, so during application design, an attempt was made to use the WordNet::Similarity tools by calling the Perl scripts from Python code. This proved complex as there were numerous out of date dependencies to trace and install.

An alternative to using a specific algorithm in a library was to use an online semantics package. Two were identified: the Cortical Compare API [46]; and the Dandelion Similarity API [47]. Both take two pieces of text and return a value representing the semantic similarity of the texts. The Cortical API was found to be better suited for comparing single word or short multi-word terms using it's `term` type [46], often returning errors when queried with two texts with a similar syntax, and in contrast the Dandelion API is better suited to single or multi-sentence blocks of text [48].

Both the Cortical and Dandelion APIs interface via a HTTP REST interface, so will be simple to implement. They are also in active development with support teams. For these reasons, they have been selected to be the semantic comparison tools used in this project.

Semantic Analysis - Discovering Named Entities in text

A named entity extraction tool is required to establish the geographical entity ('place') in a users' question. The requirements of the tool were:

1. Accepts a string of text and returns a list of keywords representing the place.

2. Keywords are categorised by a group classes, allowing for differentiation between a keyword representing a place, concept or person.

3. Accessible via a HTTP REST API for ease of implementation (fulfilling functional system requirement 10 in section 4.2.3).

During initial research, two keyword extraction API's were discovered, the Cortical Keyword Extraction API [49] and the Dandelion Named Entity Extraction API [50]. However, the Cortical API does not fulfill requirement 2, simply returning a list of keywords. In comparison the Dandelion API returns a complex data structure which includes a list of DBPedia classes that the keyword is an instance of.

Research was done to identify a way of using the Cortical API to extract keywords and subsequently categorising the keywords after they have been been identified, however as there was no significant benefit of the Cortical API over the Dandelion API this was quickly discounted and the Dandelion API choosen.

Results from the Dandelion API were checked against the `Place` class in the DB-Pedia Ontology by checking their `types` list (a list of DBPedia classes that the keyword fits into) for the string `http://dbpedia.org/ontology/Place`.

6 Implementation

This section will contain information on how data is stored, the structure of databases, and the format of and method through which data is sent to an SMS provider.

6.1 Implementation Overview

As discussed in the Method and Requirements section, the implementation was completed whilst following the iterative development process. Software development took places in stages, which represented each iteration, and after each iteration a version of the software was demonstrated to Lilian Blot in a weekly meeting. In these meetings, the software's capabilities and success rate were discussed, and requirements were re-evaluated based on progress. Ideas for expansions to the software (for example, the Natural Language Processing, which was not in the original plan) were also discussed to get feedback.

The software has been constructed in a module manner to allow easy future expansion and modification.

6.2 Tools and services used

The following tools and services were used in the implementation:

- Python was the programming language used in this project. This was due to a number of reasons, including that the author had experience with writing an application with some similar characteristics as those in this project. A full justification was given in section 5.2.1.

- MongoDB version 2.6.6 was used as the database software in this project.

- Both the application server and the database server were hosted on an Ubuntu version 14.10 "2 GB General Purpose v1" Linux server, hosted by Rackspace in London.

- No integrated development environment (IDE) was used in the development of the server application. The author choose to use textural editors due to familiarity with them, and it was felt that the assistance provided by an IDE with managing modularity/abstraction/semantic checking would not be of significant benefit, bearing in mind the size of the server application.

  An IDE was used in the development of the client application (used during testing to reduce the use of expensive SMS). This was written in Objective-C for Mac, and created in the Xcode IDE, which is the industry standard.   cite?

- The Twilio SMS Service [51] was used to facilitate the SMS interface to the server application.

41

- The Cortical [46] and Dandelion [47, 50] API's were used for semantic analysis of text.

In addition, the following Python libraries from the Python Package Index[20] were used within the server application:

1. Greenlet [37] - provides a notion of micro-threading (coroutines), which allows the server to process multiple HTTP requests concurrently.

2. gevent [35] - a coroutine based networking library for Python. Gevent provides a Web Server Gateway Interface (WSGI) server that supports processing multiple requests concurrently, using Greenlet.

3. PyMongo [52] - Python library containing tools to work with MongoDB databases.

4. ujson [53] - Twilio use JSON as the format for HTTP requests, so it makes sense to use it for all requests. ujson allows for effecient encoding and decoding of json to/from Python data structures.

5. Twilio [54] - Twilio library used for receiving and sending SMS.

6. lxml [55] - Used in the 'Wikipedia Utils' library, which handles interactions with the Wikipedia API. lxml is used for processing XML/HTML in Python.

7. Flask [33] - a micro-framework for web applications, used for it's simple syntax for defining HTTP routes (URL's that trigger Python functions). Flask also comes with a WSGI server, but it does not support processing multiple HTTP requests concurrently, so it is replaced with the Gevent WSGI server in point 2.

8. NumPy [56] - Numerical Python (numpy) is used for outlier detection in the rating system. It provides a function call for calculating the standard deviation of a set of numbers.

9. re - a Regular Expression matching library (built into Python) which is used in the 'Wikipedia Utils' library, and in application code for converting CamelCase text to Snake case text.

10. Wikipedia Utils [57] - a library used for abstracting the Wikipedia API.

6.3   Implementation of natural language processing module

In this section, a description of how the natural language processing module was implemented is given.

It was decided that a module should be constructed specifically for the parsing of natural language questions in the application. This module, called nlp, processes

---

[20]Python Package Index website: https://pypi.python.org/pypi

question in natural language form, and is able to output the two parameters needed by the application for answering a question.

The first step in the process is to remove punctuation from the question text, and this is of no meaning in the context of this application. Consider an example sentence `How tall is the Eiffel Tower?`. In this sentence, the question mark will be removed to give `How tall is the Eiffel Tower`.

The next step in the process is to identify the geographical entity that a user is investigating. This is done by passing the original question into a Named Keyword Extraction tool, which extracts a list of keywords representing entities from a string and and returns them. The Dandelion Named Entity Extraction tool was selected as in addition to providing a list of words representing entities, it categorised them according to the DBPedia Ontology. By using this, it was possible to identify the geographical entity (place) by simply checking each keyword for the type `http://dbpedia.org/ontology/Place` in the types associated with each keyword. This is done by the `extractEntities()` function in the nlp module.

The final step in the process is to identify the property that a user is requesting about the entity, for example, the height. This is done in the `stripToProperty()` function in the nlp module. This function identifies the property by assuming it will be the only text in the sentence with meaning, excluding the place name. As such, the first action of this function is to remove words relating to the name of the geographical entity been queried, which is does by checking each word in the question text against each word in the place string, and removing the word if they are equal. After this process, our example question will have been changed from `How tall is the Eiffel Tower` to `How tall is the?`. The next step is to remove meaningless words that do not identify the property that the user querying. These are stop words, such as `and,` `the,` `of`. As with removing words relating to the geographical entity, this is done by checking each word in the question text against a list of stop words and removing them if they match. In our above example, the question text will now be `tall`. This is the property determined to be the subject of a users' query.

Having called both `extractEntities()` and `stripToProperty()`, the nlp module packages these results into a dictionary and returns them as the processed representation of the question text.

6.4   Implementation of answer location system

In this section, a description of the process followed to answer a question about a property of a geographical entity is described.

After having identified the place and the property of that place in the question (using the nlp module) the application must locate the value of that property for a place. This is done in the `sourceProcessor` module, which collates data from the data sources and then attempts to identify from those sources an answer to the users question.

The entry point into `sourceProcessor` is the `findArgumentOnPage()`

function, which takes the place name and the property been queried as input parameters. The first step of `findArgumentOnPage()` is to query the data sources for a dictionary of properties about the geographical entity, pulling down all information available on the entity in question. Interfaces to data sources are factored out into modules, such as `dbpediaParser` and `wikiPageParser`, which handle queries to these data sources and return the data to the application in a normalised form (a dictionary of keys and values about the geographical entity).

After having retrieved a list of properties about the entity, the application then iterates through the datasets (one dataset per source) checking the keys against the key in the question text. If any key in the keys for a dataset is an exact (case-insensitive) match, the value for that key is returned as the answer. Else, each key in the compared to the requested key for semantic relatedness. This tries to match key strings given in different contexts, and is useful in the case that a user asks a question using a different word to represent the property they require than the one in the data source. After having compared each key to the property given by the user, the key with the highest semantic similarity is returned.

An example of where the semantic comparison and matching of keys is useful is in the question `How tall is the Eiffel tower?`, where the property string is `tall`, and in the data source the key might appear as `height`. The word `tall` has a high semantic relatedness to the word `height`, so the value for `height` will most likely be returned (unless another key with higher semantic similarity is found).

This semantic comparison is done by the the `compare` module, which has a function `similarityOfProperty()`. This function takes two strings as an input, and returns a value representing the semantic similarity of the two strings. Two services are used for the semantic comparison: The Cortical [46] and Dandelion [47] similarity API's. Both of these

The Cortical API is not configured for recognising text in CamelCase, so helper functions such as `camelCaseToSnakeCase` and `camelCaseToSpace` are also included for converting CamelCase to either Snake Case or space-separated words.

explain why had to use both, explain why using both is bad

6.5   Implementation of rating and learning systme

6.6   Discussion of Implementation

- Tweaking the constants for best results - Excluding wikipedia due to dirty data

Key issues:

- can't handle units

- sometimes wikipedia attaches other garbage

- can't do subjective stuff 'depth of the deepest ocean'

- can't do properties that are stored deep within the ontology.

7 Results

This section's content...

8 Discussion

This section's content...

9 Evaluation

This section's content...

10   Extending this project

Section not complete If the author was able to expand this project, areas for expansion include:

- **Addressing the issue of literacy**

- **Automating the expansion of the database**

- **Units for metrics.**

11 Conclusion

This section's content...

Melby reference not right, publisher weird - ask Lilian in next session. Url Commented in TEX here

References

[1] U. N. international community, "The universal declaration of human rights," 1948. [Online]. Available: http://www.un.org/en/documents/udhr

[2] H. Falaki, R. Mahajan, S. Kandula, D. Lymberopoulos, R. Govindan, and D. Estrin, "Diversity in smartphone usage," in *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys '10. New York, NY, USA: ACM, 2010, pp. 179–194. [Online]. Available: http://doi.acm.org/10.1145/1814433.1814453

[3] I. T. U. (ITU), "Key ict indicators for developed and developing countries and the world (totals and penetration rates)," Geneva, 2013. [Online]. Available: http://www.itu.int/en/ITU-D/Statistics/Documents/statistics/2013/ITU_Key_2005-2013_ICT_data.xls

[4] O. for National Statistics, "Internet access quarterly update, q1 2014 release," Online report, London, Tech. Rep., 2014. [Online]. Available: http://www.ons.gov.uk/ons/rel/rdit2/internet-access-quarterly-update/q1-2014/index.html

[5] K. Fox. (2011, Jul.) Africa's mobile economic revolution. [Online]. Available: http://www.theguardian.com/technology/2011/jul/24/mobile-phones-africa-microfinance-farming

[6] J. C. Aker and I. M. Mbiti, "Mobile phones and economic development in africa," *The Journal of Economic Perspectives*, vol. 24, no. 3, pp. pp. 207–232, 2010. [Online]. Available: http://www.jstor.org/stable/20799163

[7] I. T. Union, "Measuring the information society," International Telecommunication Union, Geneva, Tech. Rep., 2013. [Online]. Available: http://www.itu.int/en/ITU-D/Statistics/Documents/publications/mis2013/MIS2013_without_Annex_4.pdf

[8] D. Kenny, "The ethics of machine translation," 2011.

[9] G. Inc. (2014) Google research: Find out how our translations are created. [Online]. Available: http://translate.google.co.uk/about/intl/en_ALL/

[10] A. K. Melby, "Why cant a computer translate more like a person," 2006.

[11] S. Goldwasser and M. Bellare. University Lecture Notes. Accessed: 2014-11-15. [Online]. Available: http://cseweb.ucsd.edu/~mihir/papers/gb.pdf

[12] R. Anderson, *Security Engineering*, 2nd ed. Indiana, United States: Wiley Publishing, Inc., 2008.

[13] B. Schneier. (2004) Cryptanalysis of md5 and sha: Time for a new standard. Accessed: 2014-11-16. [Online]. Available: https://www.schneier.com/essays/archives/2004/08/cryptanalysis_of_md5.html

[14] C. Teat and S. Peltsverger, "The security of cryptographic hashes," in *Proceedings of the 49th Annual Southeast Regional Conference*, ser. ACM-SE '11. New York, NY, USA: ACM, 2011, pp. 103–108. [Online]. Available: http://doi.acm.org/10.1145/2016039.2016072

[15] P. Johannes B. Ullrich. (2011) Hashing passwords. Accessed: 2015-03-16. [Online]. Available: http://www.dshield.org/diary/Hashing+Passwords/11110

[16] W. W. Royce, "Managing the development of large software systems," in *proceedings of IEEE WESCON*, vol. 26, no. 8. Los Angeles, 1970.

[17] C. Larman and V. R. Basili, "Iterative and incremental development: A brief history," *Computer*, vol. 36, no. 6, pp. 47–56, 2003.

[18] B. Boehm, "A spiral model of software development and enhancement," *Computer*, vol. 21, no. 5, pp. 61–72, May 1988.

[19] K. Beck, M. Beedle, A. van Bennekum, A. Cockburn, W. Cunningham, M. Fowler, J. Grenning, J. Highsmith, A. Hunt, R. Jeffries *et al.*, "The agile manifesto," 2001.

[20] I. Sommerville, *Software Engineering*. Pearson Education, 2010.

[21] N. Rahmany. (2012) The differences between life cycle models - advantages and disadvantages. Accessed: 2015-01-27. [Online]. Available: https://narbit.wordpress.com/2012/06/10/the-differences-between-life-cycle-models-advantages-and-disadvantages/

[22] C. Larman, *Agile and iterative development: a manager's guide*. Addison-Wesley Professional, 2004.

[23] B. Boehm and W. J. Hansen, "Spiral development: Experience, principles, and refinements," DTIC Document, Tech. Rep., 2000.

[24] MediaWiki.org. (2015) Api:main page. Accessed: 2015-03-17. [Online]. Available: http://www.mediawiki.org/wiki/API:Main_page

[25] DBPedia. (2014) The dbpedia ontology. Accessed: 2015-03-17. [Online]. Available: http://wiki.dbpedia.org/Ontology

[26] ——. (2015) Virtuoso sparql query editor. Accessed: 2015-03-17. [Online]. Available: http://dbpedia.org/sparql

[27] A. Rajaraman and J. D. Ullman, *Mining of Massive Datasets*. Cambridge University Press, 2012.

[28] datatxt-nex getting started. Accessed: 2015-03-25. [Online]. Available: https://dandelion.eu/docs/api/datatxt/nex/getting-started/

[29] F. D. Davis, "Perceived usefulness, perceived ease of use, and user acceptance of information technology," *MIS quarterly*, pp. 319–340, 1989.

[30] P. community. (2015) Pypi - the python package index. Accessed: 2015-02-04. [Online]. Available: https://pypi.python.org/pypi

[31] Pymongo 3.0rc0 documentation. Accessed: 2015-03-26. [Online]. Available: https://api.mongodb.org/python/current/

[32] P. J. Eby, "Python web server gateway interface v1.0.1," 2010, this document is informally known as PEP 3333, based on it's PEP number.

[33] F. open source developers, "Flask," http://flask.pocoo.org/, 2014, accessed: 2015-03-30.

[34] A. Ronacher, "Werkzeug," http://werkzeug.pocoo.org/community/, 2015, accessed: 2015-03-31.

[35] D. Bilenko, "gevent," http://www.gevent.org/, 2015, accessed: 2015-03-30.

[36] ——. Implementing servers. Accessed: 2015-04-3. [Online]. Available: http://www.gevent.org/servers.html

[37] A. Borzenkov, "greenlet," https://pypi.python.org/pypi/greenlet, 2014, accessed: 2015-03-30.

[38] Sms delivery reports and carrier acknowledgements - twilio. Accessed: 2015-03-28. [Online]. Available: https://www.twilio.com/sms/deliverability

[39] Plivo international sms coverage. Accessed: 2015-03-28. [Online]. Available: https://www.plivo.com/sms-api/coverage/

[40] Independent states in the world. Accessed: 2015-03-28. [Online]. Available: http://www.state.gov/s/inr/rls/4250.htm

[41] A. Budanitsky and G. Hirst, "Evaluating wordnet-based measures of lexical semantic relatedness," *Comput. Linguist.*, vol. 32, no. 1, pp. 13–47, Mar. 2006. [Online]. Available: http://dx.doi.org/10.1162/coli.2006.32.1.13

[42] ——, "Semantic distance in wordnet: An experimental, application-oriented evaluation of five measures," in *Workshop on WordNet and Other Lexical Resources*, vol. 2, 2001.

[43] C. Fellbaum, *WordNet: An Electronic Lexical Database*. Bradford Books, 1998.

[44] C. Leacock and M. Chodorow, "Combining local context and wordnet similarity for word sense identification," *WordNet: An electronic lexical database*, vol. 49, no. 2, pp. 265–283, 1998.

[45] P. Resnik, "Using information content to evaluate semantic similarity in a taxonomy," *arXiv preprint cmp-lg/9511007*, 1995.

[46] Cortical, "Cortical compare api," http://www.cortical.io/developers_apidocumentation.html, 2015, accessed: 2015-03-31. On the referenced page, click "Compare", then "/Compare" for the Similarity API.

[47] SpazioDati, "Text similarity api," https://dandelion.eu/docs/api/datatxt/sim/v1/, 2015, accessed: 2015-03-31.

[48] datatxt-sim getting started. Accessed: 2015-03-25. [Online]. Available: https://dandelion.eu/docs/api/datatxt/sim/v1/

[49] Cortical, "Cortical keyword extraction api," http://www.cortical.io/developers_apidocumentation.html, 2015, accessed: 2015-03-31. On the referenced page, click "Text", then "/text/keywords" for the Keyword Extraction API.

[50] SpazioDati, "Named entity extraction api," https://dandelion.eu/docs/api/datatxt/nex/v1/, 2015, accessed: 2015-03-31.

[51] Twilio, "Twilio sms service service," https://www.twilio.com/sms, 2015, accessed: 2015-03-31.

[52] I. MongoDB, "Pymongo," https://api.mongodb.org/python/current/, 2015, accessed: 2015-03-30.

[53] J. Tarnstrom, "ujson," https://pypi.python.org/pypi/ujson, 2011, accessed: 2015-03-30.

[54]

[55] lxml open source developers, "lxml," http://lxml.de/, 2015, accessed: 2015-03-30.

[56] N. open source developers, "Numpy," http://www.numpy.org/, 2015, accessed: 2015-03-30.

[57] J. Todd, "Wikipedia utils," https://classic.scraperwiki.com/scrapers/wikipedia_utils, 2010, accessed: 2015-03-30.

12 Appendix A - Personas

12.1 Nanjala

12.1.1 Demographics

- **Female, aged 15**

- **Living in the village of Amuria, Uganda, 300km from the Capital, Kampala. Internet access is difficult and expensive to obtain in Amuria, though there is good mobile network coverage.**

- **Currently in general education at her village school.**

- **Nanjala is curious and wants to be able to ask questions and retrieve and read more information at her own leisure. She prefers to be told about something than to learn facts.**

12.1.2 Average Day

In a typical day, Nanjala will rise early to help her family with housework chores, before she prepares for school. Nanjala wants to become a teacher. At school, she studies Science, Maths and Geography, and works hard for her classes and on her homework. Nanjala's school has a small library and a body of students, all of whom are as hard-working as Nanjala, meaning that books are often unavailable when Nanjala would like to look something up. This increases the amount of time that her homework takes, and can even result in her not having access to information when she must go home to her family.

12.2 Cedric

12.2.1 Demographics

- **Male, aged 27**

- **Lives in Zurich, Switzerland, and goes on regular hiking trips, from 3-4 day trips around Europe to longer trips to the South American Andes and southern Asia.**

- **Works in a service role in the city and so has little disposable income to spend on guide books for the many places he visits.**

- **Cedric is a competent albeit not professional hiker, often going off-track and summiting 'difficult' peaks or reaching famous camping grounds. He likes to be spontaneous, only deciding which peaks he will attempt when he arrives.**

- **Cedric is very conscientious and wants to contribute his own knowledge to the world; especially to projects that help him.**

- **Cedric values his privacy is concerned about how much information he gives out online.**

12.2.2 Average European Trip

A typical trip will involve an intercity train from the Zurich Hauptbahnhof (main station) to the city closest to the range Cedric will be hiking. He takes the train because it tends to be cheaper than flights. Once in the area, Cedric will take local trains and buses to get into the mountain ranges. There, he will use the local hiking trail network signs and conversations with the locals to select a peak or route. Once Cedric has got moving and is reaching higher ground, he uses a combination of his view and the signage to identify other peaks that he would like to wander around.

Cedric takes a non-smart phone with him because a) the battery life on his smart phone is not long enough for his whole trip, whereas his non-smart phone lasts a week at a time, and b) his smart phone would not be useful anyway, since the internet is so intermittent. He still wishes he could get information like whether there are maintained routes on the slopes of a given peak or the height of a peak that he sees without having to phone friends back in the city.

Appendix B - Sample Consent Form

System to access knowledge using Machine Learning and SMS - Usability and Usefulness Study

Informed Consent - Introduction

This study aims to investigate the usability and usefulness of a system developed to provide access to knowledge using machine learning and SMS. The system aims to provide high quality answers to questions asked of it, where the questions relate to either a description or property of a geographical entity (for example, a city, river or mountain). The system aims to learn where mistakes have been made and improve by receiving feedback from the user on the previous answer.

Form - please complete this section before the study

I, _____, agree to participate in this study. I have been briefed on and understand the purpose and goals of the project.

I understand that information will be collected from me during this study in the form of my answers to a questionnaire. I understand that this information will be treated confidentially and only Sam Heather will have access to the original data in it's original format. When the information is processed, shared, described or interpreted I will not be personally identified.

I understand that I can withdraw from the study at any point without reason.

I understand that the aim of this study is to identify whether the system developed in this project is useful and efficient to use to retrieve information about geographical entities.

Date: _____Signature: _____

Researcher's contact details: Sam Heather, sam@heather.sh

Supervisor's contact details: Lilian Blot, lilian.blot@york.ac.uk

...and this section after the study

☐ **I have been suitably debriefed about the system and my performance.**

☐ **I understand I can withdraw my data from the study at any time.**

☐ **I have not been forced to take part in this study.**

☐ **The Researcher has treated me with respect and answered all of my questions.**

For the researcher: Participant ID: ____

Appendix C - Sample Questionnaire

Please fill in the following questionnaire with regard to your experience of the system that has been developed

.

#	Question	Agreement 1 = strongly disagree 3 = neutral 5 = strongly agree
1	Using the system would enable me to locate information more quickly when without access to the internet	1 2 3 4 5
2	Using the system would improve my performance when working or researching without access to the internet	1 2 3 4 5
3	Using the system would increase my productivity when performing casual research without access to the internet	1 2 3 4 5
4	Using the system would enhance my effectiveness at finding information without the internet	1 2 3 4 5
5	Using the system would make it easier for me to find information without access to the internet	1 2 3 4 5
6	I would find the system useful in day-to-day life when without access to the internet	1 2 3 4 5
7	Learning to operate the system would be easy for me	1 2 3 4 5
8	I would find it easy to get the system to do what I want it to do	1 2 3 4 5
9	My interaction with the system would be clear and understandable	1 2 3 4 5
10	I would find the system to be flexible to interact with	1 2 3 4 5
11	It would be easy for me to become skilful at using the system	1 2 3 4 5
12	I would find the system easy to use	1 2 3 4 5
13	The information returned was useful, relevant, and answered the question asked	1 2 3 4 5

For the researcher: Participant ID: _____