The Information Radar System

Niklas Johansson & Olov Johansson

ABSTRACT

As information becomes more available and mobile the cognitive limitations of man becomes evident. Sorting out relevant elements from large information space is a cognitively demanding process. In this thesis we focus on the connection between information and location to see what can be done to aid users in their tasks of finding relevant information by acquiring the user's context and location. To be able to do this, we have studied and surveyed topics related to this: How the environment and artefacts can be utilized to provide cognitive support and the areas of context awareness and mobile interfaces. From these studies we have derived knowledge and ideas that support the construction of the Information Radar, a mobile context- and location aware prototype used for scanning the vicinity of its user for interesting locations. The prototype also incorporates a virtual scribble board that makes it possible to annotate locations or other users. In this way users can add own information, public or private, to the world. The context of this thesis lies within a wider science project that goes under the name vITal Situate. This is a project driven by the European Union which goal is to make the northern part of Sweden the world leading region within the area of mobile internet.

Table of Contents

1. Introduction

Mobility and availability of information increases as the popularity of mobile computers, PDAs, mobile phones with mobile internet access increase. Large amounts of information are available anywhere and anytime. The task of finding relevant elements in the ever increasing space of information can be very difficult and calls for good cognitive tools that can help us filter out the irrelevant elements. It is realistic to assume that many users of mobile systems with internet access carry such devices to be able to get information associated to their location. For example, a personal digital assistant (PDA) can be used for examining the web page of a nearby museum to see if there are any exhibitions of interest at the moment. However, finding the relevant web page and displaying it on such a small device could be troublesome and frustrating. Another problem with this situation is that we are sometimes not aware of all interesting locations in our vicinity that we would like to visit or at least be informed about.

In this master thesis we focus on the connection between information and location. From this point we examine the possibility to develop a tool that is aware of both location and interests of the user. This tool would have the task of scanning the immediate vicinity of the user and report the existence of interesting locations, thus cognitively alleviating the user from having to perform this task. We also seek to include the possibility for the user to add information to the world by using the tool. This feature is derived from studying how we make use of location to aid our cognitive skills. For example, by placing the garbage bag beside the front door we utilize location to remind us of taking it to the waste disposal next time we go outside. Similarly, users of the tool can benefit from the possibility to add information in form of reminders or messages to the world. The task we have undertaken in this thesis is to develop a prototype of a tool that has these features. The resulting product of this task is a client-server system called the Information Radar (IR).

The IR client application is half of the fruits of this project and it runs on a PDA since such a device is easy to carry around, readily available and it was well suited to implement a first prototype on. The application itself links the virtual world to the physical world and let the user utilize the PDA as a window into the virtual world. The virtual world, from the view of this project, consists of ordinary web pages that carry information somehow connected to physical locations.

The user's position has to be sensed and reported back to the application somehow and the obvious choice, for outdoor operation at least, was a Global Positioning System (GPS) receiver. Such a device is able to track the user almost anywhere on the planet with accuracy down to a few meters and is also simple to communicate with from a programmer's perspective.

On the server side, which can be seen as the second half of the IR system, all location linked information about the physical world is stored and manipulated to present only the relevant elements to the user. This approach is also taken to relieve the client from having to carry much information since a stationary server is better suited for this task. Another reason for storing the information on a server is to make it possible for users to share information.

With the IR in the pocket a user is relived from having to search for information that is linked to its current location and can better focus on other tasks. As soon as the user enters the vicinity of an interesting area that is represented in the IR system the IR client will tell her about its existence by showing the associated web page. It is also possible for the user to add annotations in the form of web pages to the incorporated virtual scribble board by typing them into the IR client and sending them to the central server. Such annotations can be public, which means that all users of the IR can see them, or private, so that only the placer will see them. They can be placed on static locations or on other users of the system.

1.1 Thesis context

The context of this thesis lies within a wider science project that goes under the name vITal Situate, or simply Situate [vit03]. This is a project driven by the European Union which goal is to make the northern part of Sweden the world leading region within the area of mobile internet. The part of the project that lies within the framework of this thesis is focused on the development of mobile applications that are aware of its users' position, context and situation to aid in everyday scenarios. The vision behind Situate is the nomadic user – the future user that wants to conduct work, study or entertaining tasks independent of location. This requires that the information that the user needs is available everywhere and at any time. Yet, it is important that this increased availability of information does not cause the user to have to spend more time searching or maintaining the information. The approach that is used to solve this is to study the nomadic users' needs and from this knowledge build balanced ubiquitous and pro-active systems based on position, context and situation.

1.2 Related work

There has been some related work done that incorporate location- and context aware systems. The related projects of which some have worked as a source of inspiration and information to the development of the IR are presented below.

GeoNotes: GeoNotes [Geo01] is a system in which users can place virtual annotations connected to physical locations by using WLAN positioning. The annotations can be seen by other users which makes it possible to view the system as a virtual message board where the messages are connected to physical locations.

GeoURL: GeoURL [Sch04] is a webpage connected to a database containing web- pages associated with positions in the physical world. To visualize the

GeoURL system the homepage provides a clickable world map with spots marking places that have a webpage associated to them. Anyone can add their page to the database by adding a meta-tag of HTML to their web page and tell the GeoURL server the address of the page by using a simple script found on the homepage. The meta-tag contains the name of the page its physical position (longitude and latitude).

GUIDE: The GUIDE [Che00] system is a virtual tourist guide that runs on a Tablet PC. The Tablet PC is equipped with a wireless network card that in combination with a WLAN connection is used to provide location awareness. GUIDE can provide walking directions and suggest tours for tourists that want to explore a city or other interesting locations. The big advantages of using GUIDE instead of attending a normal guide session is that tourists can choose what they want to see themselves and do it at their own pace.

1.3 Structure of the thesis

This thesis is divided into seven chapters. The following three chapters can be seen as the main body of the thesis, each of these are briefly described in this section.

1.3.1 Using the Environment and Tools for Cognitive Support

Chapter two of this thesis is written by Niklas Johansson. Its aim is to provide a survey of how we can make use of the environment to aid our cognitive skills in everyday situations and how task solving can be made easier by using cognitive artefacts. This is done from an individually distributed cognition approach, which sees cognition as a distributed system that incorporates the environment and artefacts. This section also reviews the concepts of space, artefacts and environment in detail to bring clarity to these frequently used terms that are of interest to the study.

This survey serves as a source of fundamental understanding regarding the role of the environment and artefacts. It also presents some views on cognitive problems that motivate the construction of the IR that is the technical aim of the thesis.

1.3.2 Mobile Computing Interfaces and Context Awareness

The next chapter of the thesis is written by Olov Johansson and is intended as a survey of techniques that can be used in mobile interfaces. Design guidelines for ordinary desktop applications are not valid for mobile applications. Mobile devices often have limited resources in the form of screen space and processing power and several techniques can be utilized to overcome those drawbacks.

The first part of this chapter deals with some ideas for interface design and how to communicate with the user. This communication differs from the notification in that it is implied that the application already have the attention of the user. The different modalities for both input and output are covered. The benefits and the drawbacks of each modality in a mobile computing context are examined and also how several modalities could be used parallel to produce multimodal interaction.

The notion of notification is examined next, what constitutes a notification and how should it be delivered? Some guidelines on how to design notifications for use in mobile applications are covered and the also benefits of following those guidelines. Last the different modalities suitable for notification are investigated and the benefits of the different ways of delivery are discussed.

The final part of this chapter investigates the concept of context and why context is important in mobile computing interfaces. It tries to describe some different views on context and how to achieve context-awareness for mobile devices. The benefits of having context-aware applications and devices are also discussed and linked to interface and notification design.

1.3.3 The Information Radar technical specification

The fourth chapter of the thesis is the technical specification of the Information Radar prototype, a location- and context aware, mobile application for finding relevant information in the vicinity of the user. Simply put: a radar that searches for information. A feature also incorporated in the Information Radar is the virtual scribble board, which makes it possible for users to add their own information to the world. This prototype has been developed at Umeå University during autumn 2003 to early spring 2004 as a practical thesis project. The goal of developing this prototype has not been to create an application that features all desired functionality, since it is a task far too big for a master thesis. Rather, the goal has been to provide a prototype application that can be extended in the future. In this chapter all details about the prototype can be found accompanied with test runs, design issues and some visions regarding the future of the project. This is the last main chapter of the thesis and it is followed by a section in which we draw some conclusions about both the theoretical and practical work that has been done.

2. Using the Environment and Tools for Cognitive Support

2. 1 Introduction

As Einstein once said "*My pencil and I are smarter than I"*, we can become smarter by making use of artefacts (objects constructed by humans) when working. Also the environment can provide cognitive support when we are solving problems. However, this idea is not supported by the traditional model for cognitive science, since it views cognition strictly at an individual level. This model also suggests that the individual mind works similarly to a computer by processing algorithms and manipulating abstract symbols [Rog97]. Adopting this model makes it hard to study cognition in a system consisting of other beings, artefacts and tools, since the theories only consider the individual mind. More recent research [Hut95a] on cognition suggests that not only artefacts should be incorporated in cognitive studies, but also social aspects and culture. It is said that cognition is distributed among these elements, and the area of this study is referred to as distributed cognition (DC).

This is also the approach that is taken in this chapter which aim is not to present ground breaking results of research, but rather to provide a survey of studies conducted by others on the areas of distributed cognition and the concepts of environment and space. The aim is also to answer the questions: How can the environment support our cognitive skills? And how can we make use of the environment and artefacts to reduce our cognitive work load?

The following part of this survey is divided into four sections. First, some work that motivates the study of space, environment and artefacts is reviewed. Some examples are given that show how we can benefit from using tools and other things in our environment when we work. The next part gives a survey on how the concepts of space, artefacts and environment are defined, and provide some views on which different definitions exist in modern research on the topics. It also gives a detailed view on how we make use of artefacts to improve our cognitive skills. The next section deals with the main area of the study. Here we seek the answer to the question: how do we make intelligent use of space and environment? To put these studies in a wider context and to bring some background knowledge to the areas mentioned the following part reviews some history and thoughts on DC. In the end there is a summary of all the theories surveyed in this chapter.

2.2 Motives for the Study

Why is it interesting to study how we can use the space, environment and artefacts to become smarter? Simply put, all work is not easy, but it always takes place in an environment of some kind, and it often incorporates the use of artefacts. Therefore it is interesting to examine how we can increase our skills in general work situations by making use of the things around us. There are however some problems with doing this. Sometimes there are too many tools incorporated in solving a task and there might also be other elements of disturbance in our environment that makes the working process complicated.

2.2.1 Cognitive Overload

The term cognitive overload is used to denote when cognitive demands become too high for a person to handle. Modern work environments often include telephones, email, pagers, calendars, web pages, papers, books, newsletters etc. Together, their roles easily change from being efficient tools for knowledge work and communication to become nothing less than a cognitive chaos. This chaos leads to cognitive overload which in turn leads to personal stress, work discomfort and it might also strain social relations [Kir00].

Kirsh [Kir00] suggests three ways to overcome and prevent cognitive overload. One way is to change the physical layout of the work environment. This can be done by providing better tools and more free space to the worker. It might also be a good idea to relocate people in the work environment so that people with common or complementary knowledge get the chance to work in a team since it can reduce stress.

Another way of reducing cognitive overload is to change the methods, algorithms and practices that the worker makes use of. For example, upgrading the software used when working to provide a more efficient computer workplace. One could also provide the workers with PDAs for synchronizing meetings and events and keeping better track of their personal time management.

The last example also holds for the third strategy suggested by Kirsh. To provide better tools that workers can use to offload their cognitive burdens on. A PDA can aid in calculating, visualizing data, taking notes and making reminders, actions that would normally demand cognitive performance from the worker. Using good artefacts can drastically aid cognitive offloading which in turn reduces overload. Wilson [Wil02] claims that there are also situations that cannot be operated or prepared in an off-line cognitive sense. In these moments our cognitive skill may not satisfy the demands of the situation. This sometimes leads to performance failure, but not always. The environment can show up to be a good companion when our cognitive skills fall short.

2.2.2 Motivating Experiments on Cognitive Offloading

The term cognitive offloading refers to when the environment is used as an extension of our own abilities or when we use it as cognitive storage. It is said that we make use of external representations. For example, using a shopping list to enhance memory or using a calculator for solving a complex equation. There have been a number of experiments done that show that we can benefit from offloading cognitive work on the environment.

Interactive skills in Scrabble: An experiment made by Maglio et al. [Mag99] shows that building words from a predefined set of letters is easier if we can physically rearrange the letters on the table before us. A group of students were given a predefined set of letter bricks similar to those used in the game Scrabble. During 5 minutes they were to build as many words as possible from the letters and type them down on a paper. Another group of students were given the same sequence of letters and the same task, with one difference; they were allowed to physically move the letter-bricks around during the five minutes. The results showed that the latter group that was allowed to physically rearrange the letters produced more words than the group that was only allowed to use mental rearrangement.

Tetris: An experiment performed by Kirsh et al. [Kir95b] gives strength to a similar theory. In this experiment the computer game Tetris was used. The goal in Tetris is to move falling building blocks of different shapes to form complete horizontal lines when placed together on the ground. As the blocks fall to the ground one by one the player can move and rotate them to make them fit better. Kirsh et al. watched the players and noticed that rather than mentally visualizing how to rotate the current falling block to make it fit best, the player actually rotated the block "physically" (in the game). The test also showed that more experienced players rotated the bricks more than the less experienced ones. Contrary to what one may think, experienced players do not seem to build and keep mentally rotated images of the blocks as the game proceeds; they rather seem to prefer to rotate them physically since physical rotation of the shapes is up to 5-10 times faster than mental rotation [Kir95b].

In the cockpit: A study by Hutchins [Hut95b] provides observations on how pilots make use of artefacts as cognitive tools that distribute information within the cockpit. Memorizing appropriate landing speeds of the aircraft is an important task for pilots and is very important for the safety of the aircraft and its passengers. To do this the pilot that is not flying the plane first reads the gross weight of the plane from the fuel meter located on the instrument panel. This value is used to decide the appropriate speed values for landing the plane. Landing speeds for different gross weights are located in a booklet consisting of so called speed cards on which the different speed values are written. The pilot finds the appropriate card for the current gross weight and puts it on the instrument panel so that the pilot flying the plane can see it. The pilot also moves a set of indicators on the airspeed meter to indicate what speeds are optimal for landing. This study shows how lots of information can be present and used in an environment without actually being inside a human mind. The speed card booklet works as a long term memory for the pilots and prevents them from having to remember all weights and their corresponding speeds. The speed card and indicators on the airspeed meter visualize the speeds so that the pilot does not have to keep them in its memory. This study also shows how artefacts can distribute information over social structures, rather than just being a cognitive tool for one individual.

As these experiments show we obviously benefit from making use of our environment to strengthen our cognitive process efficiency. Further research by Kirsh [Kir95a] reflects over some techniques that we use in our everyday activities, not only to offload cognitive work or information on the environment, but also make strategic use of the space around us to simplify choice of action, perception and to make internal computation more effective. Before looking into that study we examine the areas of space, artefacts and the environment to provide a definition to these widely used terms that are of interest to this study.

2.3 Space, artefacts and the environment

It is possible to say that the environment consists of some type of space. This space can have different forms and attributes and it might incorporate the existence of artefacts. The aim of this section is to clarify the definitions of these three terms, space, artefacts and the environment by answering some questions: How do people conceptualize space? What is the role of the artefact in the environment and how is it used? And how do we in greater detail define the term environment?

2.3.1 Concepts of Space

Space can have many definitions; a room, a country and the universe are spaces, all of different sizes and contents. Work by Freundschuh and Egenhofer [Fre97] provides a list of different spaces derived from analyzing lots of previous research on the topic of dividing space into groups. The different categories are based on three attributes that space can have. The attributes are also mentioned by previous researchers (see [Fre97] for more about them) when defining their categories. The attributes are:

- **Manipulation**: Is the space manipulable by a human being?
- **Requirement of locomotion**: Is it necessary to move to experience the space or can the whole of it be viewed from a fixed position.
- **Size**: The size of the space compared to a well-known thing, such as a house or an ant.

From these tree attributes, space can be classified into six different categories. These categories provide, as Freundschuh and Egenhofer [Fre97] claims, a framework for how people look at and define different kinds of spaces.

- **Manipulable object space**: Manipulable spaces smaller than the human body that do not require locomotion to experience them. For example, a small table or a waste basket.
- **Non-manipulable object space**: Non-manipulable spaces that do require locomotion to experience them. These spaces are typically bigger than the human body and smaller than a house. For example, a tree or a car.
- **Environmental space**: Non-manipulable spaces that require locomotion to experience them. The size ranges from inside of houses to cities.
- **Geographic space**: Non-manipulable space that cannot be experienced by locomotion since they are too huge. Size ranges from larger-than-cities, such as counties, to the universe.
- **Panoramic space**: Non-manipulable spaces that do not require locomotion to experience them. Small or large. For example, a view in a room, a scenic overlook or a field.
- **Map space**: Non manipulable spaces that do not require locomotion to experience them. Small or large symbolic representations used to bring a reduced version of spatial information. For example, a map.

Such a categorization is needed to understand how people normally conceptualize space. It also provides, as Freundschuh and Egenhofer [Fre97] further claim, *"a theoretical basis for how for how different spaces might shape and mold the spatial cognitive representations that we create"*. This theoretical study will primarily be restricted to the first three in the list, manipulable object spaces, non-manipulable object spaces and environmental spaces.

2.3.2 Cognitive Artefacts

As mentioned in the previous section on motivating experiments, we can save cognitive computation by offloading them on external representations. One widely used type of external representation is the cognitive artefact (CA). The definition of such an artefact by Norman reads [Nor91]: *"A cognitive artifact is an artificial device designed to maintain, display, or operate upon information in order to serve a representational function."* The following part of this section will examine perspectives on how we use and benefit from incorporating CAs in our tasks and environment.

Two Views on Cognitive Artefacts

According to Norman [Nor91], CAs can be viewed from two perspectives, the system view and the personal view, both depicted in Figure 1. In the system view, the artefact is viewed from a wider perspective together with the task and the person performing it. In this view the artefact seems to extend the cognitive skill of the person carrying out the task.

In the personal view, the presence of the artefact changes the meaning of the task. The task becomes different and might require other skills than it would without the artefact. Yet, in this view the cognitive skill of the person performing the task remains unchanged.

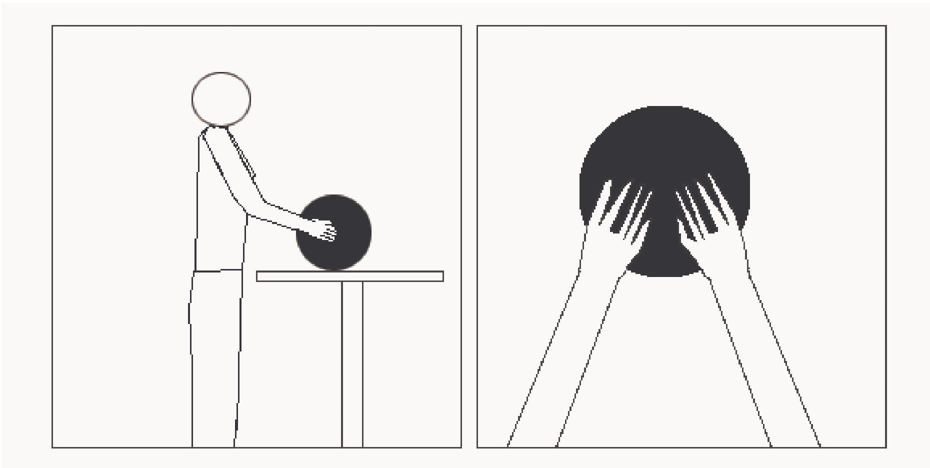

Figure 1 - To the left: the system view. To the right: The personal view

For example, an electronic calendar with a reminder function could be seen as an extension of the memory of a person. From the system view the person will appear as if she has a perfect memory and never misses a meeting. From the personal view, the task of remembering meetings will be exchanged to turning on the electronic calendar, filling in the meetings and events and then listening for reminder-signals once a meeting is about to start.

The phase in which the person fills in the meetings into the calendar is referred to as precomputation. Precomputation like this makes it possible to perform tasks when the circumstances fit us better. In this example it is possible to add a reminder for a meeting once we hear about the existence of the meeting, to prevent having to remember it all the way until the meeting actually takes place. This implies that cognitive processes can, by the use of CAs become distributed across time.

Social distribution of actions and information are also supported by many artefacts. For example, the earlier mentioned speed indicators used in cockpits to visualize appropriate landing speeds distribute information to all pilots present in the cockpit.

Using Cognitive Artefacts

When using a CA to aid performing a task, two phases can be noticed. The first one is the execution phase. In this phase we affect the world by applying an action. In the following phase called the evaluation phase, we see what actually happened to the world and compare it to what we expected to happen. For instance, using a pencil for writing text on a paper has a tight connection between execution and evaluation, as we move the pencil it instantly shows results in form of black marks on the paper.

The process is called the action cycle and is visualized in Figure 2. It is said that an artefact can be considered to support action only if it supports both the execution phase and the evaluation phase.

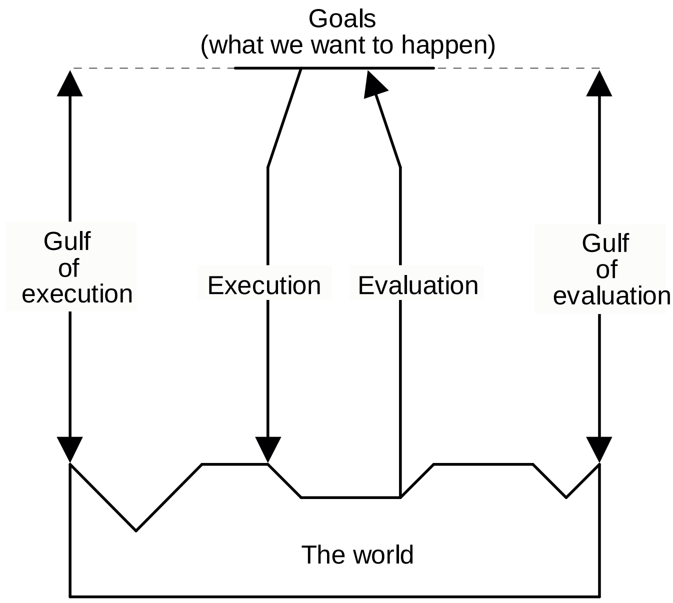

Figure 2 - The action cycle

According to Norman [Nor91] there are primarily two parts in the action cycle where problems might occur, these parts are referred to using the terms gulf of execution and gulf of evaluation (Figure 2).

The gulf of execution represents the difficulty of using the artefact and how well the artefact supports the actions used; while the gulf of evaluation denotes the difficulty to calculate the state of the environment after executing the action. It also refers to how good the artefact supports detection and interpretation of the state. There are primarily two ways of overcoming these gaps; one is through good design of the artefact, the other is achieved by practising. The goal of practising is to make the activity that involves using the artefact slip into the unconscious.

The term activity flow is used by Norman [Nor91] to denote how the activities of performing a task using a CA become effortless and are carried out without having to think about each step involved. In such an activity it is easy to feel direct engagement with the task. However, when something unexpected occurs, or something goes wrong, the agent performing the task will "flip back" into conscious control again. This can be highly disruptive for the agent and affect solving the task in a negative manner. The agent might forget where to continue the task once the disruption is taken care of, or it could totally forget about completing the task.

Disruptions are not always bad. Sometimes it is necessary to bring conscious control to the mind of the agent, since it is less error prone to action slips and not as sensible to interrupting activities. For example, when checking airplane security both pilots make use of a checklist and read each security check aloud to confirm one another. However, after several years of experience, activities like this also become automatic and more error-prone, which might result in errors, which in the case of airplane security may result in catastrophe.

To sum things up, CAs changes the way tasks are carried out and can from an outside perspective give the impression that we have better cognitive skills than we really do. Norman [Nor91] concludes that artefacts can:

- Distribute the actions across time (precomputation).
- Distribute the actions across people (distributed cognition).
- Change the actions required of the individuals doing the activity.

2.3.3 Concepts of environment

To bring the terms space and artefacts to a common context, we seek the definition of the term environment, or more specifically, the work environment. What is the definition of a work environment and what does it contain that differentiates it from our conceptions of space? Clearly the environment and space have a close relation. Kirsh [Kir99] suggests that "*Real work environments are a complex superposition of social, cultural, cognitive, and physical constraints*".

From a historical viewpoint, as the theory of DC gained more acceptance by cognitive scientists the traditional way of viewing the environment as a static external structure was also questioned. The classic definition of the environment said that it could be examined from two viewpoints [Kir99]. One of the viewpoints is called the task environment. This is said to be the part in which the outside world is located and the place where different states can be found and where actions are carried out by the agent to get closer to its goal. The other view, the problem space is the mental representation of the task environment and it is located inside the agents mind. At this point it becomes clear that this model apparently lacks aspects that the theories of DC argues to be of great importance when studying task solving agents in this fashion. Where should cultural aspects, individual experience and social aspects be placed? Inside the task environment or in the problem space?

Kirsh [Kir99] reasons that since all things we do depend on things we have done previously, the environment will clearly encompass historical properties. Also, as we do not always solve tasks individually, we cannot exclude social aspects from being present within the definition of the work environment; neither can we exclude artefacts from this definition. But how do we differentiate the environment from the agent? According to Kirsh this depends on where we set the focus of interest. For example, the fact that an agent is using contact lenses to prevent a seeing-disorder when solving a task will in most cases not be a relevant factor for the problem solving study. Unless the lenses or the sight of the agent is the focus of the study, it would probably not affect the outcome of the task in an interesting way, and the lenses would simply be regarded as an integrated part of the agent rather than an artefact that is actively used to solve the task.

In short, the definition of the environment depends on where we put the focus of interest. We cannot view the environment as a static place where only individual representations matter, neglecting cultural and social aspects of cognition. Adopting this modern definition of the environment is necessary to understand and to motivate a DC approach for studying how we make use of the environment.

2.4 The Intelligent Use of Space and Environment

We constantly make use of the space around us to store information, help us with calculations and to make it easier for us by organizing things around us. For example, we might place the garbage bag beside our front door to remember to bring it to the recycling bin the next time we step outside. When unpacking our shopping bags some of us place the frozen products in one heap, the cold ones in another and the one that goes into the pantry in yet another. We do this to simplify the coming task of putting the products where they belong, in the freezer, the pantry or the refrigerator. It is in ways like these we constantly make intelligent use of the space around us, by cueing, organizing, clustering etc.

When analyzing how an agent carries out a task from a wider temporal perspective, it is possible to notice three different areas in which the subtasks differ from each other. These areas are called long term-, medium term- and short term structuring of the environment [Kir95a].

Long term structuring occurs when we maintain our work environment to make it satisfy our demands for the specific kind of work that is normally executed within it. For example, carpenters make sure that they always have a sufficient set of tools, nails, paint etc. in storage.

Mid term structuring refers to the step when the environment is prepared for work. For instance, before a carpenter starts building a chair, she places the tools and material she needs in the particular place where the building will be carried out. Another example where mid term structuring is of great importance is when cooks prepare their work environment by arranging, chopping and dividing the ingredients, a technique also known as "mis en place": the pre-assembly of ingredients and tools.

Short term structuring occurs when the work is executed; parts of the chair are drilled and plugged together and so on. According to Kirsh [Kir95a], an agent has three reasons to rearrange its work place when the task has been started, these are for simplifying to:

1. Track the state of the task.
2. Figure out, remember, or notice the properties signalling what to do next.
3. Predict the effects of actions.

This especially applies to when expert agents work; they seem to be well aware of these reasons.

2.4.1 Expert Agents

To bring some clarification to some of the terms in the following parts of this section it is necessary to adopt some definitions found in one of the anchor papers *The intelligent use of space* by David Kirsh [Kir95a].

When referring to the term expert agent Kirsh explains that *"The agents we observe are experts, or near experts, at their tasks, despite these tasks often being everyday tasks"*. By practice the expert has learned when and where to put perceptual focus and how to categorize perceptual inputs. An expert in this sense is therefore able to find all information it needs locally, which means that it normally will not have to plan its actions on-line. To further aid efficiency, the expert makes use of structuring- and cueing techniques to prepare the environment for coming tasks. Finally, in the rest of this section the term environment refers to a work environment in which tools and workplaces are strategically pre-structured to help the agents by offering offloading capabilities for cognitive power and memory.

Expert activities can, from an analytical perspective be divided into two distinct parts, skill-based control and rule-based control. Both parts have a tight responsive connection to the environment. Skill-based control refers to when the environmental responsiveness is completely automatic. For example, an agent decides to listen to a music CD. The process of turning on the CD-player, picking up the CD-case, opening it, putting the CD into the player and pressing play does not in any of the subtasks involve reflection or planning. Skills involved in events like these can often be considered automatic and free from deliberation.

The second part, rule-based control, occurs when the expert during the execution of the task faces unexpected conditions. When this occurs the expert must fall back on its set of recovery rules to get back on track with the task. The expert makes use of such a rule without reflection. For example, if the agent notices that the CD it wishes to play is dirty it instantly knows what to do, since this has occurred many times before. Without consideration it grabs a piece of cloth and wipes the CD clean and thereafter continues the task where it was interrupted.

To summarize the activities performed by experts, it is possible to say that experts perform most of their activities in a local sense. However, some tasks are not trivial to an expert. Sometimes things get out of hand and require planning. There are also from a cognitive perspective, worse situations in which we are force-fed with too many tasks at the same time so that our cognitive skills fail us, such as in the case of cognitive overload.

2.4.2 Stabilizing the environment

As earlier mentioned, whether or not a work environment should be considered as good depends on the task; however Hammond et al. [Ham95] claims that one fundamental strategy for successful environment design is to make it possible for the agent to stabilize the environment as they perform work within it. For most people, the kitchen is a good example of a stabilized environment. We normally keep all glasses in one cupboard, utensils in a certain drawer and so on. We know where to look when we need one; therefore it is possible to say that their position is stabilized in the environment. Hammond et al. provides a list of different kinds of stabilizations that we make use of in our everyday activities; two of them have a connection to how we use the environment.

- **Stability of location:** This is the most frequently occurring stabilization technique. The above mentioned task of finding a glass in ones home kitchen is an example of how we make us of location stability to improve our efficiency. There are primarily three benefits of using an object with well known location:
    - Standard plans involving often used objects in a set location require less projection and inference.

- Plans including objects in well known locations do not need explicit checks. For example, we do not need to explicitly check that the glass is inside the cupboard before opening it.
- There is no need to search for the object since its location is by definition well known.

- **Stability of cues:** A cue is an object in the environment that affects the perception of an agent and directs its focus. Stabilizing cues is a technique that is used to trigger activation of plans. For example, a cue can be created by placing an important paper on the pillow to make sure that it will be read before going to bed. This relieves the agent from putting effort into remembering the task. This technique is also mentioned and further studied by Kirsh [Kir95a].

Kirsh [Kir95a] describes how experts can use so called jigs to stabilize cognitive processes in an environment. This basically means that experts seek to prepare the environment in a way that minimizes the need for reflection and online planning as tasks are performed within it. There are two types of jigs, informational jigs and physical jigs. They are used to cue or constrain the set of possible actions in the environment.

Informational jigs are used to minimize perceived degrees of freedom. For example, stores usually put up large signs telling customers that "This product currently has a reduced price" to attract the focus of shoppers. Doing this narrows down the shoppers' perceived number of choices they have when it comes to choosing which product to buy. In this way the informational jig works like a cue. However, informational jigs can also be used to constrain action by hiding or disabling information that for the moment is not interesting or forbidden to access. One example of such use is the intelligent menus found in Microsoft Windows XP that automatically hide items that are not used for some period of time.

Physical jigs are used to minimize physical degrees of freedom. They can be viewed as a constraining version of informational jigs. While informational jigs still make it possible to ignore the perceived constraint, physical jigs can work as a further limiting function. For example, a door can be locked to keep unwanted people from accessing a room. To take another example, in case of a car accident it is possible to make drivers choose another way by using a road block. In this case, the road block will also work as an informational jig since it is designed to be highly visible. Drivers will most likely turn around or choose another way as they see the block, not become physically stopped by driving into it.

There is one major difference between the outcome of using informational jigs and physical ones. Physical jigs can help the agent to get physically closer to its goals. Informational jigs on the other hand can only aid in an informational sense, or as Kirsh puts it, act as an oracle. An oracle in this sense is a device that is able to give the correct answer to any well-formulated question. By doing so the informational jig can point the agent in the right direction, though, it cannot help the agent to physically get closer to its goals. For example, an informational jig can tell you, in detail, how to prepare a drink, while a physical jig can make the drink for you.

Using jigs is a way of preparing and rearranging the environment as we use it. They are also used to control other agents' physical and perceived use of the environment. Kirsh [Kir95a] suggests three methods that we use to prepare the environment to make cognitive work performed in it more efficient and to simplify choice. The techniques discussed are:

- Reduce the average fan-out of actions perceived as available at decision points.
- Eliminate the need for previously necessary decisions.
- Add new heuristic properties to simplify ranking the desirability of actions.

Reduce the Fan-Out of Actions

To study how it is possible to narrow down the set of actions that seems available in a situation the term affordance is of great importance. The affordance of an object is the property of how it seems to be appropriate to use it. For example, a bed gives a good implication of a place for rest and lying down, while a hammer or a baseball bat seem more appropriate to grab and then use to hit things. It is important not to overlook the importance of the situation and the agent itself when studying affordances. A CD does not afford any listening qualities to a deaf person. In a similar way a computer does not afford any interaction possibilities if the input devices are placed out of reach for the agent. The affordance of a situation is defined as the set of actions that seems to be feasible for an agent that makes it get closer to its goals with respect to its knowledge and capabilities and how the situation itself is arranged.

Kirsh [Kir95a] defines two ways of simplifying choice by manipulating affordances of actions in the environment. He suggests that smart arrangements of the available actions in a situation are of great importance. These arrangements should both hide affordances to constrain what is seen as feasible, and highlight affordances to draw attention to what is feasible.

By dividing a task into smaller subtasks it is possible to hide affordances needed in one part of the task from the parts that do not need it. This will also reduce the available set of choices by hiding uninteresting affordances of the task irrelevant factors in each subtask. By spatially arranging the subtasks in different regions, perceptive inputs from each workspace will be reduced and it is also possible to place tools in the regions they belong instead of having them spread over the whole workspace.

For one agent working in a spatially divided workspace it is till possible to perform actions that are totally unafforded by the region if there are no environmental constraints on moving between workspaces. However, a task that is divided spatially where different agents work in fixed regions can be constrained further. By physically reducing the possibility for the agents to move between regions, unwanted cross-region affordances are said to be blocked. This removes physical degrees of freedom of the agents. Yet, this physical constraint should not be viewed as a mere limitation in the environment that never can be undone. The constraint is often there for a reason; its primary function is to work as a cue. For example, closing the door to the office cues that the person inside is working and does not want to be distracted by what is happening in the corridor outside.

Highlighting affordances are, according to Kirsh not as straightforward as hiding them. This strategy can be divided into two parts. Arranging the space to highlight the obvious thing to do and arranging it to highlight the opportunistic things to do.

There are primarily three strategies that are of use when arranging space to highlight the obvious thing to do [Kir95a].

- **Clustering**: By categorizing items and spatially clustering them it becomes easier to locate an item if its category is known to the agent, since it is located in a specific sub region of the workplace.
- **Size:** It is said that size highlights urgency. The larger an object is, the more attention it gets. This is also evident when using the above mentioned clustering technique. A large heap of objects will receive more attention than a smaller heap. This in turn makes using it appear more urgent when looking at the workplace.
- **Centrality:** Objects in the centre of the workplace draw more attention to themselves. Agents seem to prefer working with objects located in the centre, so arranging an object in the centre of the workplace will send the signal "use this" to the agent. Central objects are more likely to be used next than objects in at the edges of the workplace.

These strategies are also of interest when studying how we can highlight the opportunistic things to do. As we perform tasks we sometimes produce side-effects such as scrap material, which is often thrown away. Although, sometimes it is also a good idea too keep it, since it might come in handy as we continue working. For example, when changing the wheel on a bike it is often a good idea not to throw away the screws and the gears that held the old wheel in place. It is often better to keep them and use them to attach the new wheel later. However, as we keep and place the by-products in the workplace, it may sooner or later become disorganized and cluttered. Therefore it is important to place them so that their affordances are clearly shown. The agent should seek to cue opportunism, as Kirsh [Kir95a] puts it:

> "Opportunism is the practice of taking advantage of opportunities the environment provides in order to fulfill a goal one did not originally set out to attain -- the opportunistic goal lies outside one's current sub-goal context. Moreover, it is important that the cost to attain the opportunistic goal is lower than normal -- the context provides the agent with a golden opportunity."

Kirsh further suggests that it is important to not just place the by-products anywhere in the workplace. Three questions needs to be answered to be able to place them in an affordance-highlighting way. Which affordance will be used? When will it be used? And how should the object be placed to highlight the appropriate affordance? If these questions are not taken under consideration, the agent might miss the opportunity.

Eliminating Choices

The most efficient way of reducing the number of choices is probably by eliminating the ones that fill no function. This will reduce the degrees of freedom of the agent and clearly make the task easier, since there are fewer choices that need to be considered in solving it. However, it is necessary to eliminate choices with care, it is not wanted that the quality of the outcome is negatively affected by minimizing the number of choices [Kir95a].

An example of choice eliminating action includes drawing a copy of an image by hand using a pencil. Instead of continuously choosing where to place the pencil on the paper to make the copy as accurate as possible, a semi-transparent paper could be used. By placing the semi-transparent paper over the original drawing the need for making choices of where to accurately place the pen with respect to the original is replaced by easier follow-the-line movements.

Adding Heuristic Properties

The third way to simplify choice deals with the notion of offloading case-knowledge (heuristics) on the environment. This could be achieved by arranging the subtasks of a bigger task in temporal order. For example, spatially placing the parts of a model airplane in the temporal order in which they should be put together will reduce the number of choices that must be made in each step of the assembly. To simplify the task from a different approach, the parts can be positionally arranged and rotated to fit the position in which they are to be attached to the model. From this example it is possible to derive two distinct problems one might come across when assembling. These are referred to by Kirsh [Kir95a] as the selection problem and the placement problem.

The selection problem occurs when we have to select which part in the assembly to use next. In simple assembly tasks this can be done by ordering parts in a linear temporal order, such as in the above example. However, when assemblies require ordering of subassemblies which in turn might require ordering with respect to previous subassemblies, this technique becomes insufficient. It is then necessary for the agent to make use of pre-known structures of arrangement. For example, clustering the parts in heaps that represent their respective subassemblies and then placing the subassemblies in temporal order.

When we have made the decision on which part to use, we need to decide where and in which position the part should be attached to the assembly. This is what Kirsh [Kir95a] refers to as the placement problem. In two dimensional spaces it is hard to both encode selection and placement at the same time, since placement often requires two dimensions by itself. To solve this it is possible to use an assembly program that indicates which parts that should be connected where and when. Experts often have this knowledge beforehand and use it to prepare the space by spatially arranging the parts in a program-like way. For an example, see Figure 3.

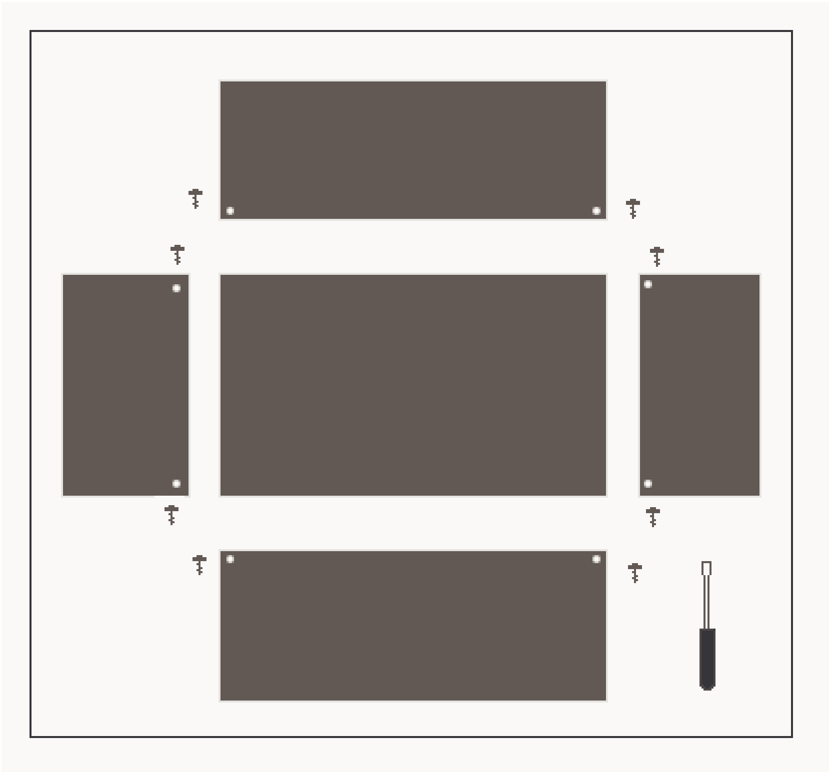

Figure 3 - An expert prepares the assembly of a drawer by spatially arranging the parts in the positions they will be assembled.

Before connecting the parts with screws, they are spatially arranged to simplify the assembly. However, this arrangement does not encode which part to use next. This could be achieved by labelling each part and supplying a scheme with temporal directives for each subassembly.

2.4.3 Spatial Arrangements that Enhance Perception

Research by Kirsh [Kir95a] suggests four ways of simplifying perception by spatial arrangements. Some of these techniques are similar to those that highlight obvious actions; however, here the aim is to enhance perception of objects.

- **Clustering:** Clustering objects in the environment makes it easier to keep track of their location and helps sorting out their affordances in relevance groups. For example, we normally do not keep our dirty laundry in the closet together with our clean clothes, for obvious reasons.
- **Dividing the workplace:** This is a similar technique to clustering. By dividing the workplace into spatial regions it is possible for an agent to

put perceptual focus at one region at a time and keep the others out of mind and sight.

- **Symbolic markings:** The purpose of using symbolic markings is to draw attention to an object and work as a reminder. However, symbolic markings lack the property of telling exactly what it is that should be remembered. For example, when cooking, a widely used technique for remembering which spoon is associated with a certain jar is to place the spoon on the jar lid when not using it to prevent using the same spoon in different jars. This technique enhances the recall process, yet it does not tell exactly what is to be remembered.
- **Clustering to enhance perceptual acuity:** By making more explicit categories of objects it becomes easier to find the functionally important elements in the environment. An example of this includes solving a jigsaw puzzle. Expert puzzle solvers tend to sort out and cluster the edge-pieces and pieces that seem to belong to a major part of the puzzle, such as the sky or walls. By making coarse categories like this it becomes easier to notice finer differences within each coarse category as we start building the puzzle.

2.4.4 Saving Internal Computation

The technique used in Tetris [Kir95b] when players offload rotation on the visual representation of the block rather than rotate it mentally is by Kirsh referred to as high speed offloading. By using the environment in this way we do not eliminate the step of computation needed to fulfil the task, we rather offload it on external representations to make it faster.

The reason that making use of external processes sometimes are faster than internal computation is mainly because we have by practising, gained better skills for acting in the external world compared to in the internal world. External representations can also be used to trigger internal ones. As in the example with Scrabble we physically manipulate the letter bricks to trigger internal computation to form new words [Mag99].

Work by Zhang [Zha90] further elaborates the notion of external representations. According to him external representations have four properties

- **They are external memory aids:** For example, using a map to find the way to a location relieves the agent from having to memorize the map and consulting the internal representation.
- **They do not always need interpretation:** They can provide information that can be directly perceived and used without explicit interpretation. Internal rules, on the other hand, require interpretation or transformation before they can be utilized.

- **They structure cognitive behaviour:** External representations can provide structures that constrain agent cognitive behaviour by allowing some behaviours and ruling out others.
- **They change the task:** By incorporating external representations in a task, the meaning of the task changes from the performer's point of view.

Viewed together, internal and external representations form distributed cognitive activities [Zha90]. The following section looks into the cognitive backgrounds required for conducting the studies presented in this survey.

2.5 Cognitive Background

As Heidegger mentions in the book *Being and time* [Hei27], as we use some technologies and tools in our every day lives, they will after some time slip into the background making us less aware of actually using them. The action of using it is transformed from being an active thought to fall back into our unconsciousness, and by doing so the technology or tool cannot be excluded from the study of cognition as it becomes an integrated part of the system []. This is one theory that is fundamental to the study of DC. Two other theories that are closely related to DC and serve as essential background studies are embodied cognition and situated cognition.

2.5.1 Embodiment

The theory of embodied cognition posits that the traditional cognitive model of the mind should be extended from being a machine for processing abstract symbols which only task is to create a mental model of the environment. Embodied suggests a view on cognition as a system that includes not only the mind, but also the body and aspects of the surrounding environment [Wil02].

Not only living beings are considered embodied. There has been a lot of discussion and arguing regarding the actual meaning of the word "embodied". Research by Ziemke [Zie00] defines five different levels of embodiment, in which agents can be divided into.

- **Structural coupling**: The very least requirement for an agent to be regarded as embodied is that it has a structural coupling to the environment. This means that even an agent without an actual body could be considered as embodied. This does not rule out non-cognitive systems. An example of this is also provided by Ziemke [Zie00] regarding a granite outcrop on the Antarctic tundra environment*: "The outcrop is persistently perturbed by the wind, and in turn perturbs the air-currents' flow. Hence, it is an embodied system according to the above definition, although* *certainly not any cognitive scientists would actually consider this an example of embodied cognition."*

- **Historical embodiment**: The result of structural coupling viewed from a historical perspective, in which an agent can adapt across time.
- **Physical embodiment**: This requires an agent to possess a physical body. The constraints on this level can be hardened by adding the rule that to be considered embodied an agent needs sensors and motors, although, this additional requirement has been discussed.
- **Organismoid embodiment**: To increase the constraints further, a body reminding of a living organism is required.
- **Organismic embodiment**: The highest requirement of embodiment demands that an agent has a living body. The origins of these restrictions can be traced back to old biological cognitive research that is further discussed in [Zie00].

As we see there exist many different opinions on what embodiment really is. Further, according to Ziemke embodiment is the concept that differentiates modern cognitive research the most from traditional cognition and much research of today regards cognition as embodied. This is also the case with the view adopted in the theories behind DC.

2.5.2 Situatedness

Another idea that is of importance for understanding DC is that of situatedness and situated cognition. According to Wilson [Wil02], embodied congnition demands for cognition to be a situated activity, which means that as a cognitive process is executed, the process is affected by the environment through perceptual inputs and at the same time the environment is affected by the agent through motor activities. For example, riding a bike is a situated activity; the biker sees, hears and feels the environment which makes it able to navigate the bike by using its body to manipulate the pedals and the handlebar. However, all cognition is not situated. There is a need for perceptual input and motor activity output for a cognitive system to be considered situated, or as it is also called, on-line. When these factors are not preset, the system is said to be off-line. Examples of such a system include planning, dreaming and remembering [Wil02].

2.5.3 Distributed Cognition

The idea that artefacts should be incorporated into cognitive studies dates back to soviets historical-cultural school in 1934 when Vygotskij [Vyg74] posited that all artefacts we use, even the simplest ones modifies the mental process that occurs when we use them, and by doing so they cannot be excluded from the cognitive model. Due to political reasons Vygotskijs paper was not released publicly until 1974. According to Rizzo [Riz04] it is surprising that it took long time for the ideas presented by Vygotskij to gain attention.

The term DC was first conceptualized by Edward Hutchins and his co workers at University of California in the mid part of the eighties. In the book *Cognition in the wild* [Hut95a] Hutchins presents an alternative view to traditional cognition suggesting that the whole area of cognition should be rethought and looked at from a wider perspective. Hutchins study suggests that cognition is in fact an advanced distributed system rather than something that could be viewed strictly at an individual level.

Another important difference between Hutchins study and the traditional study is that it was done *"in the wild"*, which means in a real environment consisting of both social aspects and artefacts, not just a laboratory environment in which focus is put on the mind of one single agent. The theory of DC focuses on how cognition is distributed between individuals, artefacts and internal and external representations.

Most research in the area of DC has been focused on how tasks are solved in work environments such as cockpits, air traffic control and naval navigation. Lately some focus has been put on how we can study DC in our everyday lives, how we use tools, our environment and space to "become smarter".

All aspects of DC are not distributed between people. Cognition is still distributed if an environment consists of just one person and one or more artefacts. According to Perry [Per99] recent research has suggested that DC can be divided into two groups: individually DC and socially DC.

Individually and Socially Distributed Cognition

Perry defines which important differences can be noticed between socially DC and individually DC and visualizes them in a table (Table 1).

As the table shows, there are points at which individually DC and socially DC differ a lot and these differences makes it possible to divide the study of DC in two separate parts. Focus in this chapter has primarily been put on individually DC and concentrated on the study of how single agents work with artefacts and how agents arrange objects residing in the environment. Some social aspects on the use of artefacts have been mentioned but not extensively discussed.

Features	**Individually DC**	**Socially DC**
Control	Centralised in the individual´s mind	Emergent, arising out of the interaction of multiple actors – no central executive
Tool use	Artefacts are used as cognitive resources	Artefacts are cognitive resources as well as mechanisms for co-ordinating distributed cognitive resources (i.e. meta-resources)
Cognitive approach	Serial cognitive process	Parallell, distributed process
Investigation and analysis	Quantative/Experimental or Functional analysis	Qualitative/Interpretative analysis
Focus	Show where representations reside and where rules or processes can be externalised in environmental constraints.	Shows the co-ordination of collaborative activities through an examination of the representations passing between actors.

Table 1 - Major differences between individually DC and socially DC [Per99]

2.6 Summary

The aims of this survey were to answer the questions: How can the environment support our cognitive skills? And how can we make use of the environment and artefacts to reduce our cognitive work load? The goal was also to examine the areas of space, artefacts and environment to bring some clarity to the meaning of these often used terms that might be of interest to the reader.

When the environment contains a lot of disturbances and elements that make working in it difficult, it is said that we are exposed to cognitive overload. It is in such situations the study of how we can make intelligent use of the environment and artefacts within it becomes interesting. The cause of cognitive overload is often that the methods that are used and available for workers are not well suited for the tasks. Also bad environmental planning and bad arrangements of the various artefacts that are used within the environment can be the source for cognitive overload.

Further, studies have been presented regarding the conceptions people normally have of space by defining some attributes that can be used to differentiate spaces from each other. It has also been shown that CAs are often used for offloading information or performing tasks that might be out of scope for an agent. A book can be used as a long term memory or a calculator can relieve an agent from having to solve mathematical problems. The CA is primarily used to change the task to an easier one, and due to this functionality the agent may from an outward perspective appear as a better task solver than it really is.

The definition of the term environment showed that rather than viewing it as a static place where only individual representations matter, it should be incorporated in the study of how we perform cognitive work within it since it encompasses external representations such as CAs that can relieve us from cognitive burdens and be used as storages for information. Also social and cultural aspects reside within the environment. However, the definition of the environment depends strictly on the task and where we put the focus of interest.

It has been shown that we can benefit from organizing and rearranging the workplace as we perform tasks within it. This is evident when studying how expert agents work. An expert agent has by practising reached a state in which it does not need to plan as it carries out certain tasks. It has also been noticed that experts often use cues and jigs to stabilize their environments. Further techniques for stabilizing the environment have also been covered. By reducing the fan-out of actions, eliminating the set of possible choices and offloading heuristics on the workplace we can improve and relive our cognitive process as we perform work.

Since the traditional model for cognitive studies does not encompass any aspects that are exterior to the human mind, it has been necessary to use another approach to explain the important role of the environment when performing cognitive tasks. It was found that modern research regards cognition as a distributed system that in addition to the individual mind also encompasses social, cultural and environmental aspects.

In the last part of chapter the model of DC and its history has been reviewed. It has been shown that embodiment and situatedness are two areas of importance to DC. In these areas the body and the environment has a more central role compared to traditional cognition. It has also been suggested that the area of DC could be divided into two distinct studies, individually DC and socially DC. The main focus in this chapter has been on individually DC where single agents interact with the environment and artefacts.

3. Context-Awareness and Mobile Computing Interfaces

3.1 Introduction

The popularity of PDAs has increased rapidly. Mobile phones are getting more and more powerful with functionality far beyond the rudimentary ability to actually make phone calls. Calendars, cameras and music playing abilities are common on mobile phones. Digital cameras and GPS receivers are also present in greater amounts. The environment where these mobile computing devices are used in differs radically from the desktop computer environment in both user interfaces and the ways the user interacts with them. Also the designers of interfaces for mobile computing can no longer assume that the user will have the ability to focus on the application since it might be used passively with the focus on more important tasks. This increases the demand for a better notification design. The application might be used in an uncomfortable environment which prohibits lengthy interaction or it might be used in a non-static setting where conditions change rapidly like a subway car where light levels and noise differ from minute to minute.

The first part of this survey will cover the user interfaces for mobile computing environments. The range of possible input and output modalities are far greater than on ordinary desktop computers but the limitations on how to display information are also harsher. Pros and cons of different input and output modalities will be discussed as well as the use of multiple modalities simultaneously to produce richer and more effective interfaces.

The task of notification is very common among mobile computing devices. A digital calendar might want to inform the user of an upcoming appointment, a mobile phone might want to enlighten the user that someone is calling or an information radar might want to notify the user of interesting objects in the vicinity. How this notification should be handled is not trivial at all, especially if there are several devices or tasks that all fight for the information fatigued user's attention. A few key points on how to design notification cues to be invasive enough to get the attention needed but not to disturb more then necessary will be covered.

One way of making smarter, less interaction hungry interfaces is to make them aware of the context they are operating in. If the application has knowledge about the situation where it is used it could adapt to the situation limiting the number of choices to a few relevant alternatives and consequently reduce the amount of interaction demanded by the user. In this chapter I will try to summarize a few views on how context can be defined from a mobile computing perspective. There will also be a section on how to collect the necessary information in order to determine the user's current context. How sensors could be used to derive contextual knowledge from sensing the physical world such as light, noise levels, temperature etc. and also how virtual sensors that can sense information in the user's calendar and e-mail for contextual purposes.

3.2 Mobile Computing Interfaces

There exist several challenges in the creation of a useful mobile computing interface. The first challenge that most developers becomes aware of is the lack of screen space. The constraints of the screen on an average PDA is only a fraction of the screen space available to desktop PC developers. Using solutions from the desktop world will most likely result in a veritable interaction nightmare. The mobile environment however have some advantages over the desktop, the modalities available to developers often include the haptic modality not commonly available to desktop developers. The input modalities also differ from the desktop equivalent. Pen or stylus-based input is superior to mouse input in some applications and using GPS as input can result in advantages that desktop computers is not likely to possess. The developer must learn to use the advantages to compensate for the harsh constraints set by the size of the device.

3.2.1 Constraints on PDA-sized Devices

The number one constraint on a PDA-sized device is probably the limited screen space. Often the goal is to access the same information as on a desktop computer but the PDA is equipped with a very small screen. Higher resolution is no answer since the physical area is the limiting factor, instead the information has to be divided into several parts and presented in sequence in order to be presented to the user. The different parts are presented in several views forcing the user to refocus his attention on each new view. Björk et al. [Bjö99] proposes that grouping of similar information the same way on different views helps the refocusing process but navigation back and forth through the information sequence may appear. The user must interact with the device to change view in order to check related information which creates unnecessary overhead so some context awareness has to be implemented to minimize user interaction.

The input is more often then not stylus based on PDA-sized computing devices and that carries both drawbacks and advantages. The input rate when using a stylus is by far slower then using keyboard. This will make text based input less attractive (entering data etc). Other disadvantages include the need for both hands to enter data and the lack of precision whilst moving. The advantage is mainly that input is not forced to be text only. There are also buttons on PDAs that most of the times are not used in any intelligent way. Since the buttons could be operated with one hand and ethnographical studies show that it is not reasonable to use both hands for input for most tasks a PDA is suited for.

Other input methods that are not standard includes tilt sensitive PDAs, key card, track points, one hand keyboards etc.

3.2.2 Input Modalities

The different modalities available for input also differ between different devices. Desktop computers use keyboard and mouse to collect information from the user. The keyboard and mouse is so common that the designer can create interfaces that exclusively use them but that is not the case when designing interfaces for mobile devices since there are no real common interfacing techniques. On PDA-sized devices there are several modalities available that not are standard among all devices. Stylus based input is for example not possible on all platforms and the designer cannot expect to have keyboard based input since is limited to a few PDAs. Input modalities for smaller devices are even more restricted since the size of the device limits the possibilities further but there are several unexplored modalities that would suite these smaller devices perfectly. A brief description of the most common modalities and some examples of how that particular modality could be used in a mobile context follows.

Button/keyboard Input

The majority of all mobile devices features input by buttons of some sort, some of the devices even have a miniaturized version of an ordinary QWERTY-keyboard. The miniaturized keyboard enables the user to enter information in much the same way as on a regular desktop computer but this also involves the risk of designing an interface that relies on the keyboard in the same way as a desktop application. The small keyboards on PDA-sized devices do neither permit the same writing speed nor the same ease of use of the desktop computer keyboard. Information entry from the keyboard should as much as possible be restricted to short inputs. There are however keyboards optimized for one handed use which can bring typing speeds close to the speeds of ordinary keyboards but these are provided as a peripheral keyboard.

Devices that contain only a few buttons can not gather information in the same way as a device that have some form of pseudo keyboard. The information gathered from the user is in this case often limited to choices from a predefined set of alternatives. The set of answers could be dynamically changed to accommodate for the users preferences but this requires that the device have some knowledge of the user’s current context. If the device contains any form of display it could show such changes in the user interface but if the device does not have the possibility to display such information the ability to modify the user interface to suit the user is limited. The extreme case of only one button on a very simplistic pen-sized device might for example be used to answer yes and no questions.

Stylus Input

Most modern PDA-sized devices have a touch sensitive display which permits input either by using the screen to draw virtual buttons which then can be pressed by the user or to use a stylus to write or draw on the display. One advantage of using a stylus is the high affordance of a pen, people instinctively know what to do with a pen shaped object and the PDA itself can be seen as an ordinary writing pad when used together with a stylus, almost everyone can understands the metaphor PDA/pad and pen/stylus. Another advantage of using stylus based input is that hand written text comes more natural then keyboard input for untrained users making the device or application easily deployed in such settings. Stylus input can also be used to enter other forms of information then text, figures can be drawn and gestures can also be recorded and used to interact with the application and there are certainly areas when the use of figures can enhance interaction. Disadvantages with stylus input are that the interpretation of handwriting is far from trivial in most circumstances. Handwriting styles often differ significantly and there are often problems interpreting letters by the same user from time to time. The letters might have to be entered in a special way to make recognition easier and this counteracts the benefits of using natural handwriting text input.

Auditory Input

Input using sound is not very common on current PDAs. There are two main applications for using sound as an input modality. The first and easier use is to sample the background noise and use the levels for enabling the application to know what environment it is being used in. If the device is used in a noisy environment the sound levels used for notification and feedback can be increased without disturbing nearby people too much. The application could also use the collected noise data to make an estimate of the location of the user and use that information to enhance the interface accordingly.

The second method is using speech recognition as a form of input. This requires quite a lot of computing power to achieve but modern mobile devices have got a fair amount of processing power and in the future some form of ubiquitous network could be used to attain the needed computing power to recognize speech. Speech recognition could in the ideal case be used for dictation of lengthy input like e-mail and similar but the possibility to enter commands by voice enables many new applications and ways to interact with existing applications. Recognizing a restricted set of verbal commands is at the moment relatively easy and is already implemented in several mobile phones for example. The use of auditory input in conjunction with other modalities are however one of the more promising applications for auditory input.

Other Input Modalities

There is a whole array of other sensors that could be used as input modalities on a mobile device. In our thesis work we used a GPS to acquire the position of the user, this enabled the user to input data by moving around and in our case the physical position of the user was the only data that actually had to be entered.

Fingerprint readers could be used to identify a user instead of ordinary pin codes or passwords, this is first of all a faster method of identifying, and a quick swipe and identification is complete. It is also more secure in the sense that no one can easily get hold of the identifying object. The size of a fingerprint reader is also considerately smaller then a keyboard so it would make sense to utilize this kind of identification for devices where both security and size is important.

As a conclusion it can be said that the available input modalities differ between different devices and the best way to design for these differ quite a bit. If the device have several available input modalities the application should enable us of as many as possible since the users current context could set demands on which that could be used. Some of the input modalities could also be enhanced if the application knew more about the context of the user and some of the input modalities could even be used as sensors to acquire context.

3.2.3 Output Modalities

Which modalities are available on mobile devices and how could they be used? This is a highly device dependent question but most devices possess one or more of the following output modalities. A brief description of the most common modalities and some examples of how those modalities could be used in a mobile context.

Haptic Feedback

The Haptic modality is a relatively unexplored area in mobile user interface research. There have been few attempts to provide haptic feedback in more then one-dimensional notification and common for the lion share of the developed applications is that they are focused on users with severe visual impairments [Ert98]. There has however been some successful research in haptic output targeting non-impaired users, for example GentleGuide [Bos03] is a system that is used to guide users through unfamiliar buildings. Consisting of two haptic actuators attached to the wrists that provides haptic feedback to help the user turn in the right direction and advise the user to stop in case of wrong turns. The haptic interface appears to be intuitive enough to be used by inexperienced users; it is easily noticed and, if redundant, easily disregarded. One other advantage with haptic output is that it is non-invasive regarding other people in the vicinity since vibration is fairly silent.

Auditory Feedback

Some form of auditory feedback is one of the most common forms of output currently available on mobile devices. The produced sound can differ in quality and complexity ranging from simple buzzer beeps to CD-quality speech, mobile phones used to offer buzzer like beeps as notification but newer phones for example offer polyphonial ring tones or even full sample playback. If the device offer stereo output several new opportunities arise. Head related transfer functions could for example be used to give the illusion of three dimensions in ordinary headphones [Beg94] by simulating the filtering effects the shape of the ears and head have on sounds of different frequencies. This could be used to increase the bandwidth of audio information by letting the user take advantage of the "cocktail-effect". The "cocktail-effect", somewhat simplified, lets the user separate a single sound source based on the physical location of the source. In a crowded room (or at a cocktail party) you can listen in on a single conversation even though there are multiple other conversations going on. The head related transfer functions could also be used to give the position of the sound meaning, the same sound from the left side has a different meaning than from the right, alerts coming from the left are related to the system while sounds from the right involves user interaction.

There are also several ways of using non-speech sound as notification [Alb98]. These can be categorized as:

- **Auditory icons.**
  Sounds associated with certain actions, this is currently the standard in desktop computing. One example is the sample that is linked to emptying the trashcan or opening a folder. The main advantage of auditory icons is that they are well suited to represent cause-effect relations between actions and objects for example indicating user initiated actions like icon clicks and notification that the PDA have been put in its cradle. Auditory icons should, if possible, be imitations of naturally occurring sounds with some form of connection to the virtual action. The technique of auditory icons has been used frequently in modern applications, sound notifications when mail arrives or a notification of when the thrash can is emptied. There are naturally drawbacks of using this method as the only auditory feedback. There is little or no possibility of mediating the relevant dimensions of the indicated action. If there are several unread mail, some of which are urgent the mail sound are not altered from the way it originally sound when mail arrives even though there are a more pressing need for the user to remember to check the inbox.
- **Earcons.**
  A language built from short sequences of synthetic tones, the sequence of the tones is then used to create complex audio messages

representing interaction or objects. Blattner et al. [Bla89] defines earcons as *"non-verbal audio messages that are used in the computer/user interface to provide information to the user about some computer object, operation or interaction"*. The main advantage is that compound earcons can easily be created by concatenating the sound sequence for the individual actions. For example the earcon for "open" and for "file" can be concatenated to represent the action "open file". If the earcons are designed correctly about 80% of them could be correctly interpreted by the user of the system [Bre92, 93]. The drawback of concatenating earcons is that the playback time is increased, the solution is to use parallel earcons, which means that several earcons are played back at the same time taking advantage of the human ability to process parallel information.

- **Sonification.**
  Sonification is well suited to represent time-varying multivariate data, background processes, and transient conditions [Sca94]. Sonification essentially uses data to drive changes in the qualities of a sound. Most common is to manipulate the amplitude, frequency and duration of sounds. This approach is useful for exposing small changes over a wide possible range of data values [Sca94]. The drawback with this approach is however that there are no natural connection between the sounds and the data they represent [Sca94]. This method is commonly not used in consumer applications but can be found in scientific applications like astronomy or in system surveillance etc.

- **Hybrids.**
  There are also hybrids of the above mentioned methods. One of the more successful hybrids tries to combine the advantages of auditory icons and sonification and thereby nullify the disadvantages. Albers describes a method of combining the two to produce a system which is good for cause-effect relationships in time-varying data, easily understandable and identifiable sounds and provide a well defined strategy for manipulating the presented sound [Alb98]. This will reverse the two main disadvantages, artificial sounding tones (sonification) and the inadequate strategy for manipulating the presented sound (auditory icons). The solution presented for this is to divide the operational state of a specific subsystem into three different states (normal, warning and critical) and give each state a different variation of a specific sound. In Albers example the power system have a motor sound as representation, a smooth running engine for the normal state, a sputtering engine for the warning state and a choking engine for the critical state. The system then defines which sound that is most appropriate to play back.

Visual Feedback

Visual feedback may appear in many forms ranging from a single flashing LED to detailed animations on high-resolution screens. The lack of screen space or even the lack of a screen altogether might pose problems of using the visual modality for presenting information to the user. Typical desktop applications focus on presenting large amount of information to a user relatively focused on the application, this is however not normally the case with mobile applications. Mobile applications have to be more selective when displaying information to the user visually. If the device lacks a display all together it might even be impossible to display the raw data, for such devices, the visual modality is better suited as a notification channel with other modalities conveying the bulk of the information. For example, a flashing led on the user's pen-sized computer might notify the user that a new e-mail has arrived and that the email might be viewed at the nearest computer terminal just by placing the pen in the pen identification reader on the screen instead of displaying the whole message on the pen itself. A PDA-sized device may display more visual information then a pen-sized device but there is still a big difference between a PDA screen and a desktop screen. The amount of information must somehow be condensed to fit the medium and there exists several interesting techniques for this.

The technique of focus and context visualisation actively tries to display contextual material connected to the current focus of the user. One application of this technique is the Powerview application [Bjö00]. In the Poweview application, the screen is divided into one main area and several smaller context areas. The main area displays a brief version of the current foci information and the smaller areas all display related information. The information in the context areas can all be selected and moved to the focus area and the context areas are all refreshed with new information. On a desktop application all the information could be displayed at once but on the smaller screen this might not be possible so only the current focus and immediately related information is shown. The context sensitivity is the main difficulty of using this approach, all the contextual information has to be hard coded into the information or linguistic context have to be derived from the focus information, something that currently is quite hard to do.

A short summary of output modalities in mobile computing devices can be that there is generally more diversity then in desktop computing. The output modalities that differ most from the desktop environment are perhaps the visual feedback. The limited screen area can often be a hindrance and must be overcome somehow. The output modalities can also be used more efficiently if the device or application has knowledge about the context where it is being used. This can be exemplified by a screen changing its contrast values according to changing light conditions.

3.2.4 Multimodalitiy

Multimodality implies that several modalities are used at one for input or output. Why should multimodal input and output be used in mobile applications? There are several advantages with using several modalities at once, several channels of information increase the bandwidth and fluidity between interfacing humans and machines. By using two or more modalities at once the theoretical amount of information that could be transferred is increased and the amount of information that is possible to process by the user becomes the new upper limit [Joh97]. Different situations sets the level of information processing capacity available and which modalities available. Redundancy in modalities can also be a persuasive argument for using several modalities. If the user is for example driving a car down the highway and paying close attention to the surrounding traffic a device using the visual modality may be insufficient to attract the attention of the driver, a device using both visual and auditory signals have higher probability of attracting attention

Multimodality is often seen as an output specific area, but the advantages of using multiple modalities in input are far from insignificant. The possible rate of input is often a limiting factor when dealing with mobile devices, PDAs for example use gestures as the primary mode of input but the average input rate is only a fraction of the rate a touch typist can achieve using an ordinary keyboard. The PDA is however not limited to that single modality, research focusing on using gestures combined with speech recognition to achieve higher input-rates and higher input-accuracy have proved successful [Joh97]. The redundancy of the two modalities proved to be very useful, whilst on the move the speech recognition could be used as primary input, since gesture input requires high accuracy, and when noise makes speech input difficult gestures could be used as main input source. One of the modalities could also be used to correct errors made when inputting data on another modality, if the speech recognition system detected ambiguities it could provide a clickable list with choices for the user. If the gesture recognition is used in conjunction with the speech recognition it could also aid recognition, if the user says triangle but draws an object that as could be interpreted as a circle the system would still be able to draw the correct conclusions.

The drawback of using multimodality in user interfaces is that the design of a usable interface becomes more complex:

- The designer not only has to design for several modalities, the designer must also design for the use of all modalities at once which poses several design challenges. If the modalities are to support each other the designer must know how this is accomplished, this more often then not comes down to knowing the current context of the user. The system can then make decisions based on this context but the designer is still responsible for connection certain actions to certain choices.

- If the modalities should cooperate in a redundant fashion this demands a well planned design, how can the system decide which modality that should be the primary and which modalities that should supportive? How should the interface for both primary and secondary modalities interact? How should the system handle ambiguities?
- How should the system handle situations where one ore more of the modalities are unavailable? How does the transition from multi to single modality affect the user? If the same system is used on another device with different modalities available how are the user interfaced best mapped to retain consistency?

To conclude this section it can be said that the design of interfaces for mobile computing differs in several ways from the desktop equivalent. They have to be designed for the harsher restrictions set by the devices size and also the tasks the device are used for and the demands that task carry. Both input and output modalities differ from device to device and also differ radically from the desktop case. Which modality to use in the case of several available modalities is best determined by the context of the application. Modalities could also be used in conjunction with each other to produce more reliable interfaces that provide redundancy.

3.3 Notification

The difference between ordinary user interfaces design and design for notifications are that the ordinary interfaces can assume that they have got the users attention. This is however not the case for notifications, they can be designed with from the standpoint that the user is currently engaged in other activities or at least that the user is not paying attention to the device.

With this fresh in memory imagine a multitude of ubiquitous devices trying to deliver status reports and notifications all at the same time to its user. If that user is also involved in tasks that require full or even partial attention the information overload produced by the invasive devices could pose a real problem. Some of the notifications could very well be important enough for the user to shift his attention in order to interpret the notification; some of the information could even be crucial for the task the user is performing but there is also a substantial chance that the information is redundant or not important enough for the user to care at the time and as a result just increase the users stress level. The amount of information the human mind is able to process at a given time is restricted and the job of choosing which information the user should receive and how it will be conveyed is not at all a simple task. Tarasewich [Tar03] proposes meta-information to limit the information load for the busy user. With meta-information Tarasewich means the information needed to describe the real underlying information. Meta-information will decrease the amount of severe interruptions by making the transmitted information easier to process for the user. If the meta-information is deemed interesting the user may opt to view the real information. For example: A driver of some form of motor vehicle may not want to be notified as soon as a message from an arbitrary device is available, that may even be distracting and turn the focus away from operating the vehicle. Instead the driver might want to restrict what devices may notify her when she is performing a specific task. The messages should be expressed as meta-information like source and importance so that the driver can decide if she wants to be informed of the whole content of the message or if the information should be saved for later.

The device not only has to compete with the users current task there is also the possibility of several information devices present at the same time. Most information devices have similar ultimate goals, to communicate some form of information to a user. If there is only a single device the job of informing the user without interrupting is straightforward enough but as soon as there are several devices that tries to communicate all at once problems starts to arise. The different devices need to be aware of each other and able to decide in which order that should address the user and how to give notice in the best possible manner. How arbitrary devices should communicate with each other is an enormous problem that needs to be solved, how to solve it is not clear but the use of internet could be a good partial solution. The devices all need to have some connection to the internet and protocols for communication between these devices need to be developed. The problems of using Internet for communication and synchronization of devices are mainly connectivity, bandwidth limitations and battery power consumption.

Another solution to this problem may be to have smaller personal networks that consist of a predefined set of devices. This will solve the problem of knowing which devices that is in the vicinity since the application can do a swift attempt at contacting known devices. The system could also only try to contact the suitable devices since the system knows the capabilities of all devices in the network. The network would know which input and output capabilities that are present.

The second and possible even bigger problem is to determine which device that would be suitable for communication with the user, and the preferred communication method. For example if a phone call is received while the user is occupied by a very important meeting the preferred notification should not be the mobile phones ring signal. A more graceful method of notifying the user in such a situation is either through some haptic feedback (vibration) or possibly displaying a message on a screen on some device in the network such as a clock or on a PDA. The system should know which devices that was in use in the meeting so it for instance would not display a message on a laptop that was used for presentation or a PDA that was not active at the moment. Another example might be how to display a received e-mail. There are several ways to go about this task. First the user could plainly be notified by an auditory, visual or haptic signal and required to use another device to access the e-mail but the content of the e-mail might give several more options of how to display it to the user. If the e-mail just contains a few lines of text scrolling the message across the display of a clock might be sufficient or if the environment allows it the message might be read aloud by a device capable of translating text to speech. However if the e-mail contains more information like pictures or a great amount of text a clock display would be a poor choice but the system might still use it to display a summary of the message so that the user could read the message in length if it appears interesting and a device capable of displaying a lengthy message is nearby.

Tarasewich describes information cues as *"the precursors to further information retrieval on mobile devices"* [Tar03]. This includes how and when the user should get notified as well as by which device in the ubiquitous case described above. The notification cues can be visual, auditory, haptic or multimodal and can be of different urgency such as a subtle vibration to more obtrusive methods such as a loud auditory signal. Tarasewich further describes four important guidelines for designing a notification cue.

- **Getting the intended recipient's attention.**
  First of all the information cue must be able to attract the attention of the intended recipient. The user might be in crowded and noisy environments and the notification needs to be able to successfully make itself noticed in the surroundings. There is a fine balance between being to subtle for the user to notice and too obtrusive that it drown out other information. If a notification is used so frequently that the user stops paying notice to it the notification designer has also failed in his task of getting the attention of the user. The context of the user is the main factor that controls the appropriateness of the notification and also the chance of successfully catching the attention of the recipient.
- **Clearly conveying information to the recipient.**
  A notification also has to deliver some form and level of information. In the simplest case the device which delivers the notification is also the source of the information. In more complex cases the notification also has to convey the source of the information in the notification. The context of the recipient is the determining factor here too. There are tradeoffs between the amount of information that is possible to deliver and the context of the user. If the user is in a quiet environment like a library haptic feedback is the obvious notifier but the amount of information vibration for example can communicate is severely limited. The amount of information can also be limited by the use of multi-

layered cues where an initial notification sends one dimension of information and the user may choose to receive a second dimension of information. Multi-layered cues can be useful in high-stress environments where the information load is high and the designer wants do avoid increasing it further by transmitting unnecessary amounts of information to the already burdened user. Multi-layered cues are essentially meta-meta-information, informing the user that there is meta-information available and the user must opt to receive it.

- **Minimizing the disruption of the recipient's current task.**
  Even though the notification should attract the user's attention it should not severely disrupt the currently performed task unless it is of critical importance. Again the context of the recipient is important; the system should in one way or another be aware of the user's current task and adapt the notification-scheme to the task. The designer must be aware of the cognitive loads involved in the performed task and design notifications that do not conflict with them. If for example the user is involved in listening for irregularities in a sound-stream an auditory notification might not be the best choice since it probably interrupts the listening more then a visual cue. If the current task demands complete attention cues that convey a smaller amount of information might also be preferred to high bandwidth cues.
- **Minimizing the disruption of other people in the vicinity of the recipient.**
  There is a trade-off between the obtrusiveness to nearby people and the ability to attract attention from the intended recipient that depends on the social context. If the user is located in a library the amount of auditory force that can be used is strictly limited but the same notification could use significantly more force in a crowded subway car. There might also be important to let people in the vicinity know that you have been notified in some way or another so that perceptions of rudeness can be avoided. Tarasewich suggest that the guidelines for social situations involving new technology will probably evolve in the near future and help create more socially acceptable notification cues.

In short it can be said that the purpose of a notification is to attract the users' attention but not disrupting the user when the current task is more important then the information. The user might have several devices that are competing for attention and in the ideal case these devices should cooperate. The devices might queue their notifications in order of importance and notify the user in turn. Not only that, but the devices should also clearly convey their information leaving no room for misinterpretation by the user. And finally, the notification should try to minimize the disturbance of nearby people.

3.4 Defining Context

What is context? The definition of context varies greatly between different fields of use. Most often location is thought of as the main factor of context but in a mobile computing environment the context can better be described as the different variables that alter the use of an application. These factors might for example include physical, psychological, social and environmental factors.

Why is it important to know context? Simpler and more efficient interfaces can be designed if the context is known. For example if tasks of the user is known the way of presenting the can be modified: If the information that is to be transmitted from the device in question to the intended recipient can give clues of how to do this most efficiently. If the information is an alarm to indicate that lunch is about to start a complex animation requiring a laptop is not as efficient as a simple auditory signal from a mobile phone or a clock. A second example showing the benefits of knowing the location of the user: If the user is in a crowded environment a discrete visual signal might not be appropriate but the same signal might be enough in a study hall setting.

Quite a few models describing context have been developed to this date. Among the first were Schilit et al. who described context-awareness as a system that have the capabilities to examine the computing environment and react to the changes in that environment [Scd94]. Their environment included location, nearby people, hosts and accessible devices. This expands the traditional view of context from merely location to both location and co-location.

Expanding further on their theories Schmidt et al. presents a practically complete model for context for mobile computing [Sci98]. A context according to them can be summarized by:

- A context describes a situation and the environment a device or user is in.
- A context is identified by a unique name.
- For each context a set of features is relevant.
- For each relevant feature a range of values is determined (implicitly or explicitly) by the context.

They organize their feature space in a hierarchical fashion with two broad categories at the root level, human factors and physical environment. Both of these categories are then subdivided further and those categories are even further hierarchically subdivided. All categories are also affected by time all this can be seen in figure 4. The human factors are subdivided into user which contains information about the user (age, sex, habits, emotional state etc.) social environment which includes co-location, social interaction, group dynamics etc.

and tasks that consist of spontaneous activities, engaged tasks, general goals etc. The physical environment contains conditions that incorporate both relative and absolute position as well as co-location. Infrastructure describes surrounding resources (communication and computation for example) but also task performance and last the physical conditions which encompasses noise, light, pressure, acceleration, temperature and so forth.

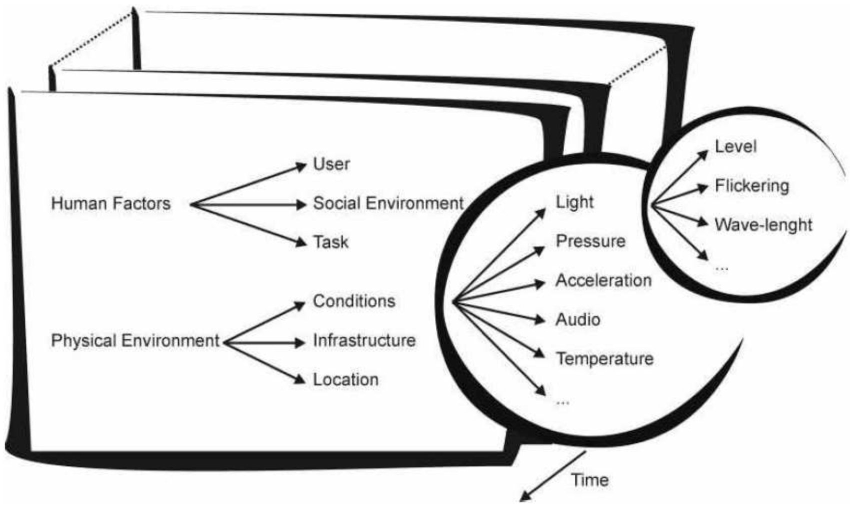

Figure 4 - Context Feature Space Model according to Schmidth et al.[Sci98]

In a parallel work, Schmidt proposed a second more general model [Scd98]. This model uses a three-dimensional feature space (see figure 5) to describe an arbitrary context. The relevant dimensions are: environment, self and activity. The environment-axis consists of both social and physical environment. The self-axis includes device state, physiological state and cognitive state and finally the activity-axis deals with behaviour and task. One drawback of this approach is that it is harder to separate a point in the context-space since a single feature could appear on several of the axis.

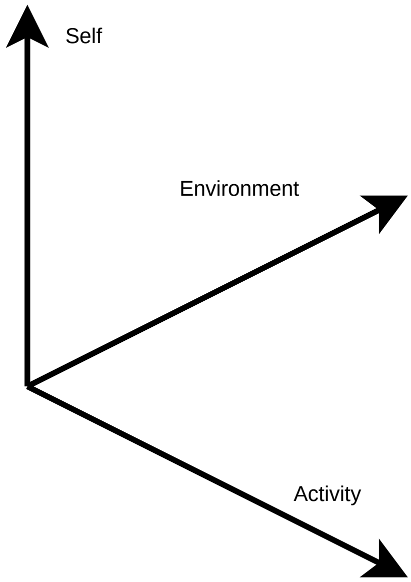

Figure 5 - 3 Dimentional Context Feature Space Model according to Schmidth et al.

Other researchers have tried to extract other minimal sets of features that can be used to describe the context of a user. One other set of such features have been proposed by Abowd and Mynatt [Abo00]. They posit that context can be summarized by the minimal set consisting of the five Ws':

- **Who**?
  Not only the user using the application but also other people in the vicinity of the user.
- **What?**
  Interpretation of what the user is doing either through automatic information gathering or by the answers of questions posed to the user.
- **Where?**
  Where the user is located and where the user is going.
- **When?**
  The current time and also the elapsed time in a specific context. Used to discover deviations in usual patterns.
- **Why?**
  The most challenging question of all, why the user is performing a certain action.

3.5 Context-Awareness

A system that is able to measure relevant context factors and create a model of the user's current context and then use that model to enhance the overall functionality of the system is said to be context-aware. A context-aware system would be able to modify functionality according to the user, environment, activity, time and goal of the operation. However the context of the user is a complex question that has to be addressed to design mobile applications better adapted for everyday use. Van Laerhoven and Aidoo writes [Lae01]:

> "the notion of context is very broad and incorporates lots of information, not just about the current location, but also about the current activity, or even the inner state of the person describing it. As a consequence, people can describe their contexts in different ways, even if they are in the same location doing the same things."

The vision is to achieve simpler interfaces through context-awareness, interfaces that do not require abundant interaction with the user. According to Chevrest et al. [Che01] the three main ways of simplifying user interfaces are:

- **Simplifying/reducing task specification.**
  Reducing the amount of input/action from the user. Automating as much input as possible.
- **Changing the output produced by the system.**
  Reducing the quantity of information that has to be processed by the user or increasing the quality of the information presented. Summarize information and remove unnecessary information that clutters the application.
- **Reducing the complexity of the user's mental model.**
  The user should not if possible have to interact with difficult unfamiliar concepts. If possible map the underlying logic to a more familiar form this is most often achieved through the use of intelligent agents that perform some of the user's tasks.

In our thesis project the use of active input from the user have been reduced by basically use the physical location of the device for input. This is achieved through a GPS receiver that tracks the user's movement and periodically updates the information bound to that location. The user does not have to input his location at any point and does not have to interact with the device in any way to receive the information connected to that location. There are other examples of devices of this kind that can use other context than location for automatic input such as time of day, temperature or height above sea-level.

Reducing the quantity of information presented to the user can be solved merely by a different design choice. The designer could choose to show less information to the user, only showing the most vital parts or she could let the user do those choices by the way of configuration. Semi or fully automatic methods could also be used to decrease the sheer amount of information. Techniques like text reduction and context and focus visualization can be used with some success.

Reducing the complexity of the mental model required to operate the application can be done in a large number of ways. If for example the application requires the user to manipulate several maps of a city to find out where different historic locations are to be found and then cross reference these locations with a historic database to find out more information in order to plot a tour through the city. A way to reduce this task would be to suggest a context sensitive tour by using the current location of the user combined with the user's interests and the proximity to the destination where the user has his next appointment.

There are nevertheless some problems related to context. Chevrest et al. [Che01] defines three problems:

- **The problem of failing to reach a stable state.**
  If both the user and the system try to adapt to the same context the end result might turn up completely wrong. Both the user and the system might end up counteracting each other. The user might perceive the system as unstable and unpredictable. Designers of context-aware system must take in to account the amount the mental models of the system the user may already have formed by interacting with similar non-context-aware systems. Expert users of similar systems can have a harder time to adapt to the new mental model the context-aware system represents.
- **The trade-off between prescription and freedom.**
  These kinds of systems sacrifice some of the user's freedom on behalf of easier interaction and simpler feedback. There is often a balance to be found between these two opposite poles, more experienced users might need the ability to perform more advanced input at certain times, this can be created through configuration and options to override the context-awareness.
- **The user must trust the agent performing adaptation on his or her behalf.**
  If the user does not trust the system to perform operations on his behalf the system will never be any success. The user must be able to trust the system to adapt to the context without interference. The user must be able to trust both the systems ability to correctly sense the context and also trust any intelligence in the system making choices for the user.

3.5.1 Acquiring Context

How should a system go about the task of acquiring context? The obvious answer is that it depends on which aspect of context that is to be gathered. Different context-aware systems require different information to function properly. To exemplify using Abowd and Mynatt's five Ws [Abo00]:

- **Who?**
  In a mobile computing environment the user is more often then not is known by the device. A device is often constantly used by the same user, a shared device could however require some form of login to identify who currently is the user. If the device is part of a distributed network of some kind the device and its user may have to identify repeatedly and this will increase the importance of correctly understanding the context of who is using the gadget and to whom or which services the user wishes to identify.
- **What?**
  Identifying what the user is about to do is one of the most daunting tasks. This can be achieved through the use of a mix of different sensors perceiving the ongoing situation; the resolution of the context picture provided to the system depends on the sensor types and quality. Sensors involved would perhaps be proximity sensors that can sense presence and distance to other humans or audio sensors that can record if the user is engaged in verbal communication. The sensors could be part of the device or part of a ubiquitous environment that provides information to requesting appliances. A second method of finding out what the user is up to is to analyze what is happening in the application that is in use and try to recognize what the current task is. This approach has already successfully been used in applications on ordinary desktop PCs and the same approach could easily be used in mobile environments.
- **Where?**
  The location of the user is often regarded as the most important aspect of context especially in a mobile context. There are already several ways to determine the physical position of a user. GPS have successfully been used for positioning in several applications including our own and both the mobile network and WLAN networks can be used to pinpoint devices on the move. Other ways of locating users that may be more relevant in some circumstances include proximity sensors, camera based solutions and active badges. [Bru01]
- **When?**
  Probably the easiest context aspect to determine but to useful it may have to take several other aspects into consideration. First of all the current time may indicate what the user should be doing, a simple

cross-reference with a calendar may provide enough information but more advanced methods that scan e-mails for appointments and analyzes conversation may be possible in the near future. Secondly the time spent on a task may give hints on how the progress is coming along. If a user spends alarming times browsing for files the file browser might be configured wrong for the user so presenting an option for reconfiguration may be the best next move in other task providing help can be the best option.

- **Why?**
  Perhaps tied with "what" in difficulty, figuring out why the user is performing a certain action presents a complex problem. This side of context can use knowledge from other aspects to build a model of the user's background, by knowing when a user is involved in a task the system might acquire clues on why the user is acting that way, if the user is supposed to start a meeting and is working the menu on his mobile phone the system might draw the conclusion that the user is about to turn the answering service on and help the user to quickly navigate to the right menu. If the systems know that the user is in a meeting room there is even stronger evidence that a meeting is about to take place and that the answering service is supposed to be turned on. If the people present are the same people that are supposed to be at the meeting the system can be even surer that the reason why you are flipping through the menus of your mobile phone is to turn off the sound and turn on the voice mail.

3.5.2 Sensors

As can be seen from the previous example the main development needed to design systems that is able acquire context is sensors of different modality. There already exists sensors capable of capturing some of the most important modalities but there is a need to develop these sensors further. If Moore's law can be applied in some level on sensors as well as on other computer components the future indeed looks bright. Sensors essentially needs to be deployed in greater numbers to be useful, a complete set of sensor modalities needs to be inexpensive enough to include in future devices. For mobile devices there is also a need for these sensors to be as unobtrusive as possible.

In the above examples there are several cases where sensors are of paramount importance. In the who-case the user could securely identify by using biometrics. Fingerprint readers are nowadays common among top of the line PDAs and in time this method of identification can become a feasible way of identification, replacing PIN-codes for mobile phones etc. When a user is identified by a correct fingerprint the device would automatically both know who the user is and if that user has the privileges to use the device. Other sensors that may be used for the same task are active badges [Bru01] that transmits the user's identity to nearby devices or cameras that could utilize video recognition to determine who is present in a certain area.

Sensors that can be used to handle what the user is doing include a number of modalities. First of all proximity to other people can be used to give clues of what activity the user is taking part in. Proximity sensors often rely on sonar using ultrasound signals that are emitted and the returning echoes are timed to provide distance information to other people. Infrared sensors that pick up heat emissions from people and objects could also be utilized, these would however not give any information on distance to the objects. The proximity to others often needs to be used in conjunction with additional sensors to achieve a positive guess on the current task. Microphones could be used to probe the sound levels of the current area and help build the systems model of the current action. By cross-referencing the users calendar virtual sensors could be created that can give input to the systems model of the current action.

Sensors for that can help describing the uses physical location are already available in the form of GPS-readers which communicates with satellites in orbit around the planet. These devices can tell the position of a user with precision down to a few meters, longitude and latitude as well as the height above sea-level can be deduced. Drawbacks with this method of positioning are that the sensors only work outdoors where a clear line of sight can be drawn to the satellites. Other possible methods to determine location of users and devices includes various methods of using the mobile phone networks to triangulate the users position, a service already available in some mobile networks. WLAN positioning is a third way of using already present equipment to locate users; this is a viable option to position users inside buildings where GPS might have trouble connecting with enough satellites. The active badges already mentioned can also be used inside buildings to provide information on user whereabouts but this approach requires expansion of the existing infrastructure. Redundancy in these systems may provide a seamless transition between indoor positioning and outdoor positioning.

There are a multitude of other sensors that could be used to help the system build a correct context-model including temperature sensors, heart rate monitors, altitude sensors, accelerometers, light sensors etc. The device in question often only needs a few of these to provide a context-model with enough resolution to provide enough context-awareness to make the life of the user easier but in general redundancy in sensors offers the systems a more complete model. There are also possibilities for several devices to share their information with other devices and by collaboration provide a better context-model. More expensive and computationally more demanding sensors could be built into buildings and shared among all present devices, for example a mobile phone might not have the computing power to perform video recognition and high quality cameras are still to expensive and large to be embedded in consumer hardware.

To exemplify using Abowd and Mynats five Ws:

- **Who?**
  A mobile device might already have knowledge on which person that is using it since mobile devices tend to by rather personal but a sensor that could be used for identification of the user is a fingerprint reader.
- **What?**
  This is the hardest task but using the user's calendar as a virtual sensor might be a good start. If the calendar is up to date and detailed enough the application might derive knowledge
- **Where?**
  This might be sensed by some positioning device for example GPS or WLAN positioning.
- **When?**
  Most devices have built in clocks that can be utilized for determining the time.
- **Why?**
  A difficult question to answer by any application but as with the what-case the best approach might be to use several different sensors to acquire the necessary parameters do decide this. A clock together with a positioning device might want to venture a guess a about the users goals with entering the bathroom late at night.

3.6 Summary

The definition of context has more or less been restricted to geographical location in the past but this is not enough for modern applications. In order to create context-aware applications that effectively adapts to the situation more then raw geographical knowledge has to be measured. Abowd and Mynatt defines the minimal set of knowledge to describe context to the answers of who, what, where, when and why. Schmidt et al. describes context in a hierarchal fashion with human factors and physical environment at the root level. Schmidt also defines context as a point in a three dimensional space with environment, activity and self as the spanning axis.

When context has been properly defined the next task in constructing a successful mobile computing interface is to acquire the relevant variables. This relies heavily on sensors connected to the device or application that senses the variables that makes up the relevant context. Sensors can either collect information from the physical world around it like noise levels, ambient light or temperature or they can be sensors that senses virtual information like information extracted from e-mails or from the users calendar. When the acquired variables have been used to construct a context model that model have to be used in order to produce some form of context aware interface.

Notification is intimately related to the context since successful notifications is dependant on knowing the context. Tarasewich describes four guidelines for designing good notification cues. The notification should be able to get the intended recipients attention, the notification should be just forceful enough to attract attention but should not annoy other people or disturb the user in more vital tasks. The notification should clearly convey its information, after all notifications primary task is to convey some form of information. This is also a delicate problem since the amount of information a notification is able to convey is closely related to the context. A notification should also minimize the disruption of the user's current task unless the notification is of higher priority then the current task. And finally the notification should always minimize the disruption of other people in the vicinity of the recipient.

The constraints of an application for a mobile computing device are far more restrictive than those on a desktop equivalent. Most mobile devices are lacking in screen space and processing power but the advantage is that the devices often possess modalities not available on desktop computers. These advantages have to be exploited in order to compensate for the disadvantages.

Input modalities common on PDAs include stylus input which has the advantage that it allows for non-text input, figures and gestures can enhance a user interface. The drawback with stylus input is that it is relatively hard to use whilst moving and that handwriting recognition is still quite hard to perfect. Buttons keyboards can be used on mobile devices but in most cases lengthy text input is a bad idea since input speeds are limited. Buttons tend to be one of the best choices for minimal interfaces since they demand little physical space are easily implemented and generally works rather well. Auditory input has the potential of becoming an important modality for mobile devices, in order to replace textual input speech recognition have to evolve quite a bit but as single word commands and as a backup channel in multimodal input it is already possible to use.

Output modalities are however more like the ordinary output channels on desktop computers, the only modality that does occur more frequently on hand held devices are the haptic modality and that is not able to convey any large amount of information. This leaves the visual and the auditory modality as the two possible channels for larger information communication. The visual modality is somewhat restricted due to the size of the display but it can successfully be used. Focus and context visualisation seems to be a successful technique when dealing with small displays. The auditory modality is perhaps the most common one for notifications but it can also be used to convey larger amounts of information. Some techniques discussed were auditory icons which generally is used for notification, earcons that could be used for more comprehensive auditory feedback concatenating smaller parts to form longer and more information-dense sentences. Sonification could also be used to represent time-varying multivariate data and hybrids of all these techniques can be used to overcome weaknesses.

Multimodality has also been briefly discussed mainly as a technique to enhance and support another modality, gesture inputs accompanied by voice commands to boost recognition performance. It can also be used for redundancy; if one modality is temporarily unavailable another can serve as the main input. If the user is walking and stylus based input is difficult the user can still give voice commands and if the user is using the same application in a library voice commands might not be appropriate so the application will solely rely on the stylus based input.

4. Technical Review of the Information Radar System

4.1 Introduction

The IR is a context- and location aware system that strives to relieve its user from having to search for relevant information in the increasingly available and mobile space of information. It also incorporates functionality resembling a virtual scribble board for placing public and private location associated information in the world that can serve as reminders or publicly distribute information to other users of the system.

The goal of the implementation of the IR is not to provide a system that includes all desired features that could be incorporated based on the theories we have mentioned so far in throughout this paper, since it would be a task far to big for a master thesis. Rather, the goal is to develop a prototype that can be extended in the future.

The aim of this chapter is to present the work behind the IR. In the following sections all parts of the IR system are described in detail and various test scenarios are presented. Some ideas on how the system can be extended in future versions are discussed along with the connection between practise and the theories presented in chapters two and three. Some design issues that we have come across during the development are also presented. In the end of the chapter we draw some conclusions regarding the results of the development process.

4.2 System Overview

The goal of this project is to develop a cognitive tool for retrieving relevant virtual information about elements in the vicinity of the user. It is important to make sure that using the tool changes the task of finding information to an easier one. Runtime maintenance of the tool should be minimized to prevent unnecessary interaction that might annoy the user more than help it.

The tool should be as small as possible and yet be able to bring feedback to the user when close to an interesting object or location. Preferably haptic feedback should be used, since auditory feedback can be hard to perceive in noisy environments in which the tool might be of great use. For example, on an exhibition, in a marketplace or in the midst of a bustling city square.

To realize this some pre-requisite work needs to be done. First of all, it is necessary to provide and store a mapping between physical locations of all objects and interesting places and their virtual representations. For example, saving the longitude and latitude of a location in a database along with the URL to the web page the location is associated with would provide such a mapping.

It is also necessary for the users of the tool to provide a set of interests that they want to be informed about. This is done to prevent the tool from turning into a nuisance that provides irrelevant and uninteresting information. Such a tool can easily turn into a cognitive burden and become a cause of both cognitive- and information overload, which is highly unwanted. The information about physical locations that will be presented to the users is manifested in the virtual world in form of web pages, images and other shared virtual objects. Therefore the users' interest profiles need to be stored virtually too, so that the system can make relevant decisions based on comparisons between information of locations and the users interests.

This structure of information and interests demands that the tool itself has a connection to the virtual world, in this case the internet. It is also necessary that the tool is aware of its current location. This is also what sets the demands for the possible hardware that can be used to realize the idea, which is presented in the following section

4.2.1 Client Overview

The client it the part of the system the ordinary user have contact with. The two main functions that it provides are the information radar ability. It helps to keep lookout for any interesting objects in the user's vicinity and if there should be any such objects nearby the client produces a summary, notifies the user and presents the summary. This will cognitively aid the user in the task of searching for relevant information. It can also extend the users knowledge of interesting sites by informing about locations that were unknown to her before using the IR. The second main part of the application, the virtual scribble board feature, enables the user to annotate her surroundings or other people in the system. These annotations can then be picked up and read by other. In this way users can make private notes for themselves that can work as reminders, or socially distribute information by making the annotation public.

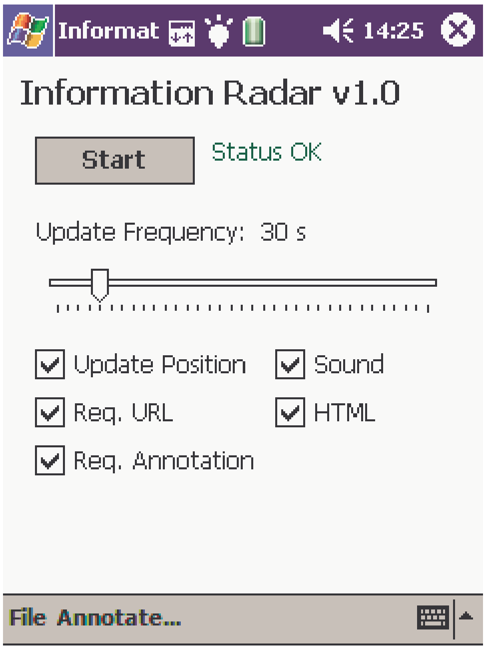

Figure 6 - The client's main window

The main window (figure 6) of the application allows the user to select which of the features of the program that should be enabled. Update position allows the server to keep track of the user, Request URL will enable the client to keep a lookout for interesting URLs in the vicinity, and Request annotation keeps a lookout on the virtual scribble board for nearby annotations. The last two let the user configure how she will be notified by the system.

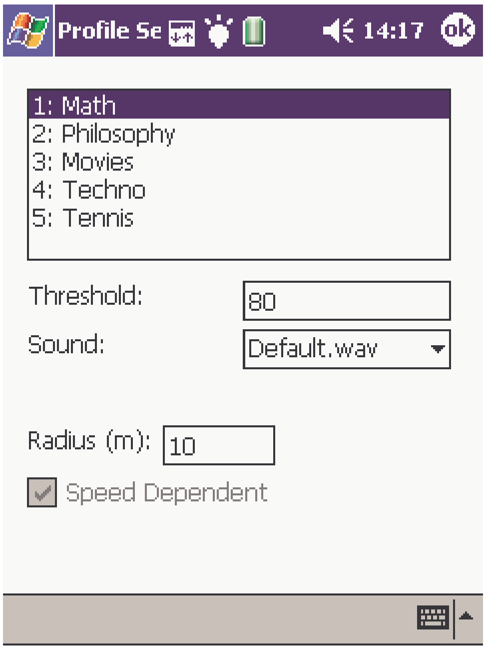

Figure 7 - The configuration form for profiles

In the profile settings screen (figure 7) the user can configure the behaviour of the client. The user can select an interest profile and set the minimum required relevancy for notification. The user can also configure which sound that should be associated with the profile so that it can easily be recognized without having to look on the summary screen. The radius of the circle centred on the user in which the client will look for information can also be set. By marking the checkbox "Speed dependent", the radius will automatically adjust itself depending on the travel speed of the user and the poll interval for requesting new information. This feature sees to that no information hotspots are missed when moving at a high speed.

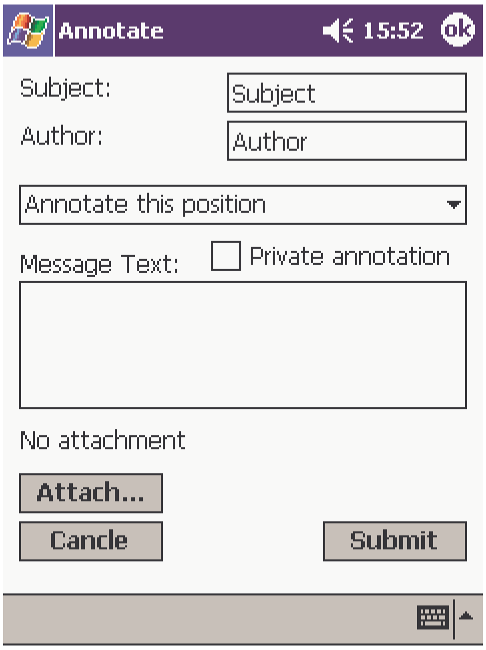

Figure 8 - The annotate screen

The second main part of the client application, the virtual scribble board, is to enable users to annotate their surroundings. This can be done by the application and as seen on this screen (figure 8) you can select the alias you want to use when annotating, also the subject that will be shown on the summary screen can be altered. There is also the possibility for user's to select what they want to annotate, persons or geographical locations. The user can also make a comment private if she likes. Finally the user can select files to attach to the annotation, primarily pictures and sounds since they are easy to display in the other users browsers.

In the end of the development a separate branch of the client was introduced to accommodate the need for a Windows version of the client. The functionality in this version is basically the same with only minor differences, a few in the user interface and a few others in the configuration possibilities. The user interface differences are mainly the location of the menu bar and the text box input for threshold instead of the slider in the Pocket PC version. The Configuration files have options to select which COM port to use for the GPS receiver since this could not be detected automatically as in the Pocket PC version. One example of how the Windows version looked can be seen in Figure 9.

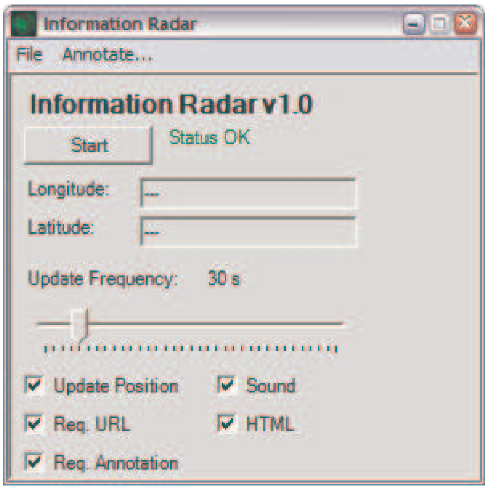

Figure 9 - The Windows version of the client

4.2.2 Client-Peripherals Layout

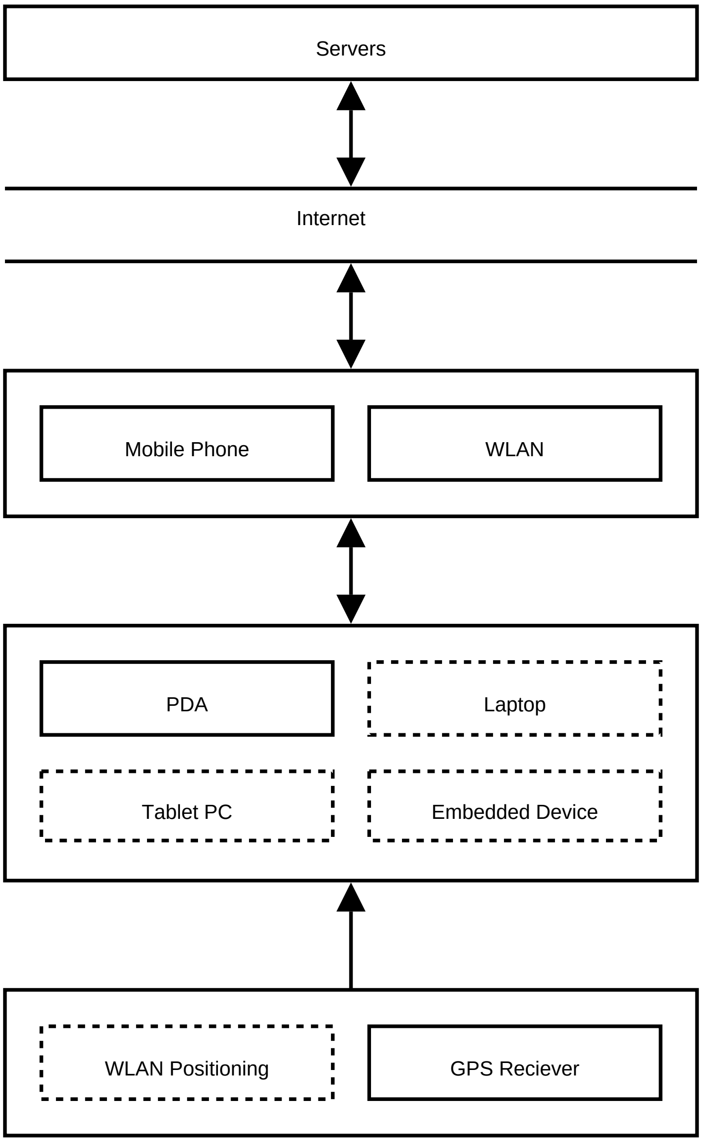

Figure 10 – Client-peripherals layout

The current client is written for PDA use but there is no real effort in converting it to run on any platform that supports .NET there are in fact already plans on converting the program to run on Windows XP. In figure 9 the basic layout of the client and the peripherals can be seen. The dotted lines are things that are not yet implemented. The client receives its geographical position from a GPS connected to the PDA by Bluetooth [blu01] but any other serial connection is possible to utilize. The use of other devices for positioning is also possible, the most feasible at the moment is WLAN positioning but there are problems with that solution on the Pocket PC Operating system. To access the internet any preconfigured internet connection will do, during the development we have used a mobile phone connected to the PDA by Bluetooth [blu01]. The mobile phone in turn communicates with the Internet by GPRS and is able to transport information to and from the servers that way.

4.2.3 Client-Server Layout

The system consists of several different servers and one central database. During this master thesis we have developed one of these servers namely the Information Radar Server (IRS) which is the systems central server closely connected to the IRS is the Information Radar Database (IRD) which contains all the relevant metadata in the system. The IRD is in our case a stock MySQL [mys95] server but the use of MySQL in particular could easily be changed allowing any ODBC compatible database to be used in its place. These two servers can be physically located at different sites as long as both have access to the Internet. The second custom built server, the Relevancy Server could be distributed to yet another computer. This server is at the moment merely a mock-up server that provides no real functionality since it just returns random values instead of real relevancy ratings for the provided profiles. At the moment the construction of this server is relatively open, it might need access to the IRD but a design that lacks this requirement is possible. The last two servers are as the IRD ordinary FTP and WWW servers, in our project we have used PureFTPd.

FTP Server

This is an ordinary FTP server where each user that is able to annotate has write permission. When a user annotates a physical location the client uploads the annotation material (html pages, images, sounds etc) to the FTP server. The current client does not take already existing files with the same name in to consideration, they are overwritten by default. The FTP server should be connected to the Web Server somehow so that the uploaded annotation material can be accessed from the web by a well known easily derive address. The current incarnation of the system uses a PureFTPd [pur01] server that lets each user upload filer into a web accessible directory, one such directory for each user.

Relevancy Server

This server is in charge returning the relevancy for each received profile for each received URL. It should only return URLs which have relevancy ratings higher then the given threshold. This server is currently not implemented since it is out of this master thesis scope. We have however made a dummy server which returns random relevancy ratings for each given profile and URL.

Information Radar Server

This server is the central server in the whole project, it processes the majority of the commands from the client and is the only server which manipulates data in the IRD. The main tasks for this server is to keep the IRD table of user locations updated, return URLs with physical manifestations in given areas (this also includes annotations), keep users profile preferences in order and bind annotations to the physical space. The IRS primarily uses the IRD as a source of information and the communication between the two are through standard SQL.

Information Radar Database

In our system this is a standard MySQL database [mys95]. This contains a number of different tables described in Appendix B. The only way to access this database is through the IRS.

WWW Server

This server is used to host annotation information. In our case it is an Apache web server [apa95] which serves the content of the directory to which the user uploads annotation information. For example if the user annotates a physical location with the image `landscape_photograph.jpg` and uploads it into her directory `/example_user` on the WWW server located at www.serveraddress.org then the address will be:

```
http://www.serveraddress.org/example_user/landscape_photograph.jpg
```

4.2.4 The database

All information regarding users, information, virtual scribble board annotations and profiles is stored in the IRD. Here follows a description of the different tables in the database and their purpose, for more detailed information on tables and their corresponding values, see Appendix B.

User: Here all information about the users is stored. Each user is associated with a unique numerical id that can be found here. This table also carries each user’s name and location.

URL: In this table all locations that carry information are stored. Longitude, latitude and the radius for each location together with the URL to the webpage can be found here. Each URL also carries a title describing the information it contains.

Annotation: This table contains the coordinates and radii of the annotations placed on the virtual scribble board. It also contains the URL to each annotation body, the title of the annotation and the signature of the author. If an annotation is placed on another user the coordinates in this table are not used. Instead, a field containing the id of the annotated user is used so that the location of the annotation can be read in the user table described above. There is also a field that holds the id of the user that placed the annotation if the annotation is private. If the annotation is public this is marked by setting the user id to a negative value, since all user ids are positive.

Profile: This is a rather small table containing three fields, a name, a unique numerical id and the id of the user that owns the profile. In this table all profiles in the system are stored.

4.2.5 Communication

Since the application does not have any information regarding physical-virtual connections it must somehow communicate with a database that keeps this kind of information on file. For this project we have developed a network protocol for just this purpose. The set of operations that we have implemented contains commands for:

- Requesting and setting annotations using the virtual scribble board, information with a physical connection added to the virtual world by the users themselves.
- Requesting information by supplying the physical location of the user. This will enable users to enquire if there are any virtual points near the user's physical location.
- Requesting relevancy for a certain virtual point. The user can query the system on approximately how interesting a certain virtual point is given the users interest profile.
- Requesting a list of other users in the system. This will enable the user to check which users that are online and using this system at any time.
- Update the user's position. Sending information on where the user is located to the server for tracking purposes.

Annotation Request / Annotation Response

The client sends the annotation request to the IRS which in turn queries the IRD then returning a list of annotations whit physical areas within the requested area, the information flow can be seen in figure 10. Se Appendix C for more information on `ANNOTATION_REQUEST` and `ANNOTATION_RESPONSE`.

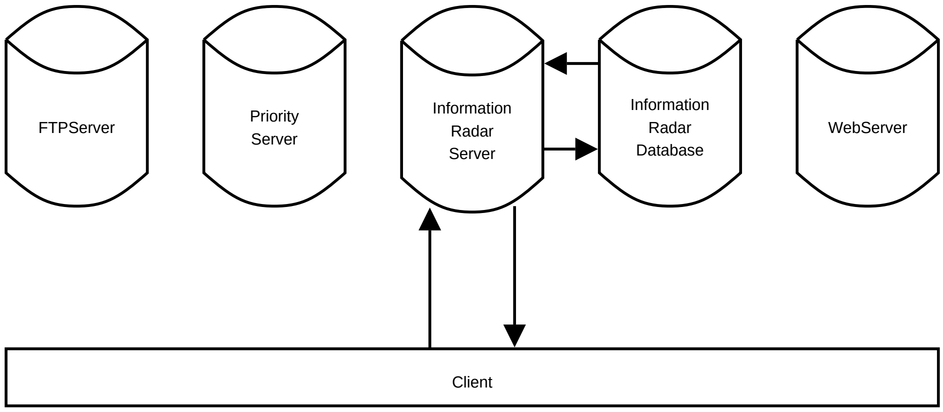

Figure 11 – Information flow on annotation request/response

Profile Request / Response

The client sends a Profile Request to the IRS. The server queries the IRD and returns a list of names and profiles the user is currently a member of, the information flow can be seen in figure 11. See Appendix C for more information on `PROFILE_REQUEST` and `PROFILE_RESPONSE`.

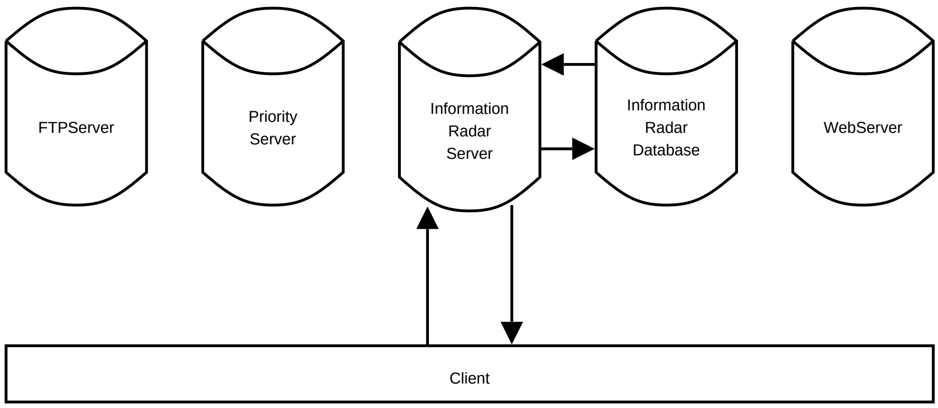

Figure 12 – Information flow on profile request/response

Set Annotation

This command is used to add an annotation to the virtual scribble board. The client uploads the annotation content to the FTP Server and then reports the URL of the newly created annotation to the IRS along with position and size of the physical area. The IRS enters this information into the IRD. The FTP server should be connected to the web server so annotation content immediately becomes accessible from the web, the information flow can be seen in figure 12. See Appendix C for more information on `SET_ANNOTATION`.

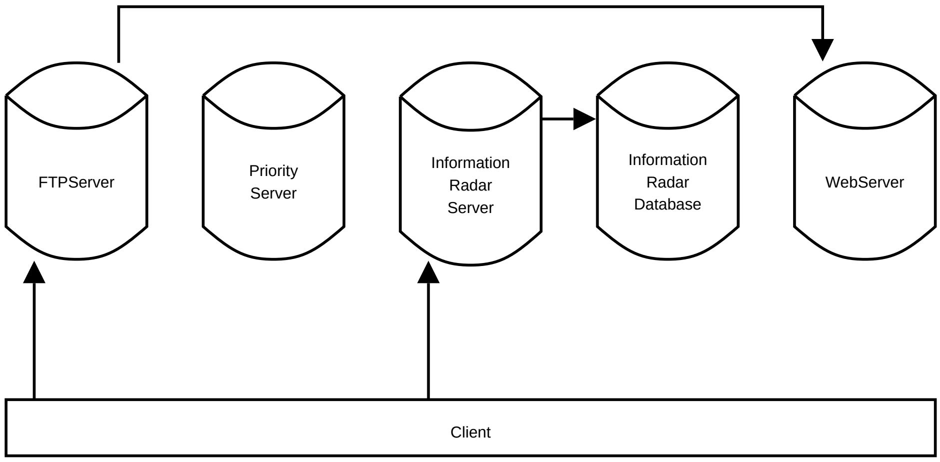

Figure 13 – Information flow on set annotation

Update Position

The client sends an Update Position command to the IRS containing user ID and position. This information is updated in the IRD, the information flow can be seen in figure 13. See Appendix C for more information on `UPDATE_POSTITION`.

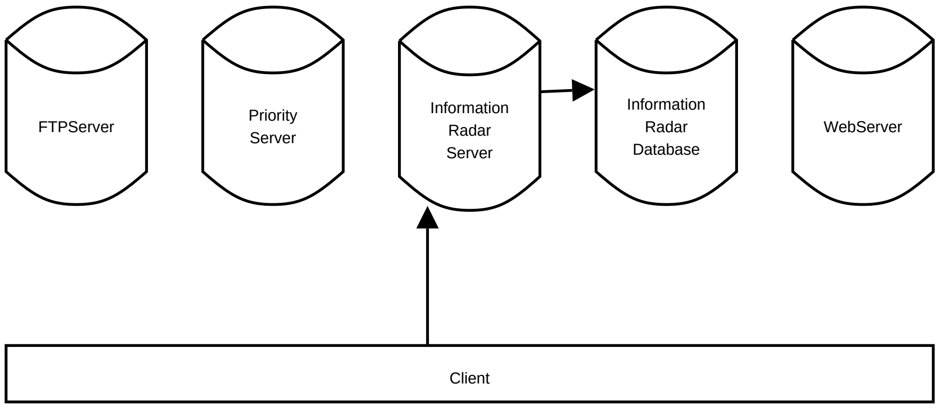

Figure 14 – Information flow on update position

Url Request / Url Response

The client sends position and area for a position to the IRS. The server queries the IRD and then returns a list of URLs that satisfies the request, the information flow can be seen in figure 14. See Appendix C for more information on `URL_REQUEST` and `URL_RESPONSE`.

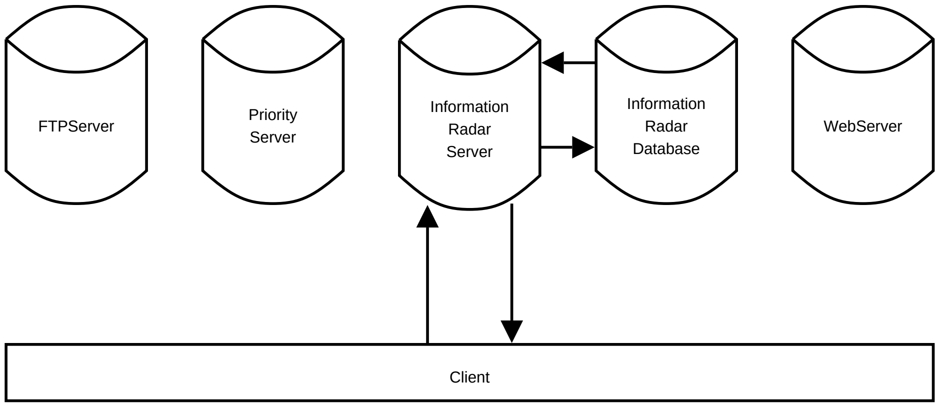

Figure 15 – Information flow on URL request/response

Userlist Request / Userlist Response

The client sends a request to the IRS. The server queries the IRD and then returns a complete list of all the users in the system, the information flow can be seen in figure 15. See Appendix C for more information on `USERLIST_REQUEST` and `USERLIST_RESPONSE`.

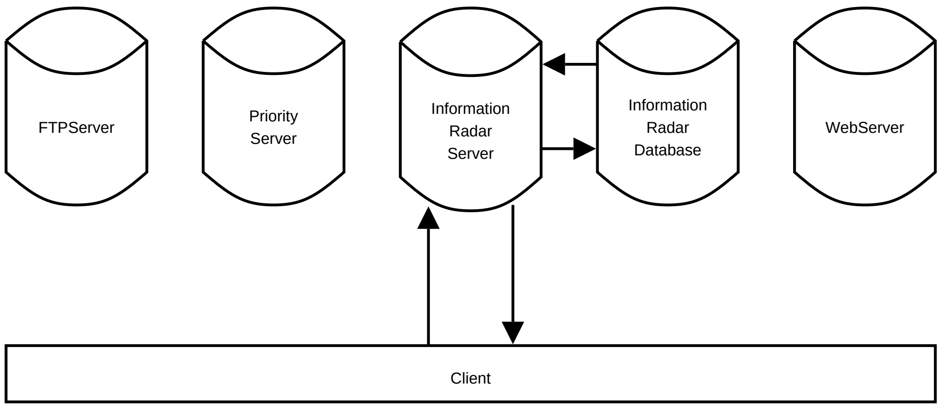

Figure 16 – Information flow on userlist request/response

Priority Request / Priority Response

When the client has a list of URLs in the proximity it sends a Priority Request for these URLs to the Relevancy Server. Along with this it sends a list of profiles and threshold values that the Relevancy Server uses for evaluation. The Relevancy Server then performs some calculations and then returns a list of URLs with associated profiles and relevancy ratings, the information flow can be seen in figure 16. See Appendix C for more information on `PRIORITY_REQUEST` and `PRIORITY_RESPONSE`.

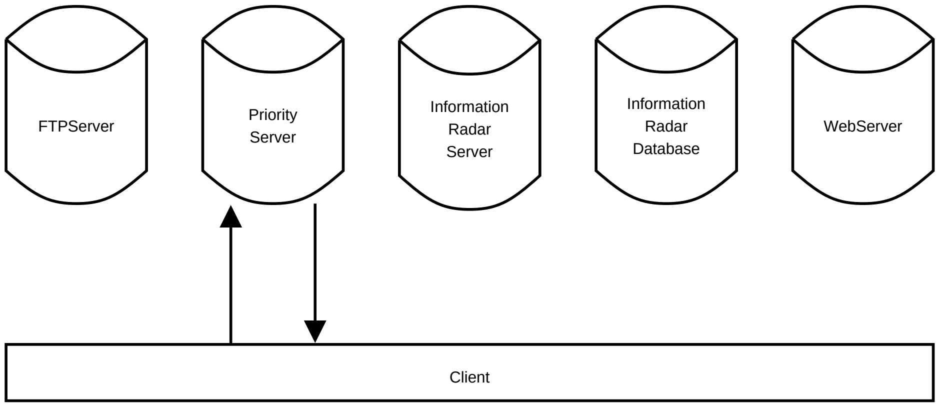

Figure 17 – Information flow on priority request/response

4.3 Development Process

The first thing that had to be done when starting this project was to decide which platform and hardware to use. Umeå University offered us to borrow a popular and advanced PDA: the iPAQ h5550 that offered both WLAN and Bluetooth connectivity. Bluetooth would be required to connect to a mobile phone that provides internet access, and WLAN could prove useful in the future to add a WLAN-positioning feature, so the choice of hardware for the client was easy. After looking into which different platforms for development would be possible we stumbled on to the .NET Compact Framework which in combination with .NET C# seemed as a good choice for developing Pocket PC applications. To the current date we have not regretted these choices.

When the first version of the IR project was finished the system looked quite different from what it does now. Many parts have changed since the development started, some have been added and some discarded. The first version consisted of the IRD, the IRS and the client. The client could make requests for profiles and web pages but nothing more. The plan was also to include the relevancy request functionality in the main server.

Public text annotations with attached images and sounds were introduced in the following version of the prototype. It was first planned for a standalone program called The Virtual Scribble Board, but we soon decided to include this as an annotation functionality in the IR. Since storing lots of files in a database is not very efficient, file transfer functionality was built-in to the server. As files were sent to the server they were stored in a folder on a computer in the same network as the server or on the same computer as the server were run. The files were then shared by running a web server and the URLs were stored together with the annotation details in the database. For some time the server also supported pushing the files directly onto a remote web server, although, this functionality was soon discarded due to security reasons.

Thoughts arose on dividing the server into two parts to make the system scale better. This is when the priority server came into the picture. Since the real server dealing with relevance scores for URLs is not finished while working on the prototype, a simple fake priority server was developed that responds with random priorities for each URL sent to it.

Instead of using the file transfer function to the server we decided that it would be better to use available standards for file transfer, namely FTP. This would also relieve the server further by making the client just sending it the URL of the annotation which is saved as a HTML formatted file and also uploaded using FTP to a folder located on a public web server. If a file was attached to the annotation it was simply linked within the body of the annotation message. This technique makes it possible to show attached images directly in Pocket Internet Explorer and also use it to download sounds and play them automatically with Pocket PCs built in audio player.

Further extensions were also added to the annotations. Up to this moment, annotations were always public, now it was also possible to make personal ones working like reminders. Another idea also arose, to not only being able to annotate locations, but also other users of the system for both public and private use. For example, one can add the annotation "Talk to me about the thesis paper" to a person, and as soon as that person is online and in the vicinity the client-program will give a reminder to the person who placed the annotation.

4.4 Test runs

All features available in the prototype have been tested in a real life environment in the campus area of Umeå University. In this section the results of the tests are presented. The testing session starts between the MIT building of Umeå University and the restaurant NH (Nationernas Hus). Both buildings have entries in the IRD. The user in this test has two interest profiles, food and school. When performing an URL request in the between these buildings with the radius set to 20 meters, the server returns the result shown in figure 17.

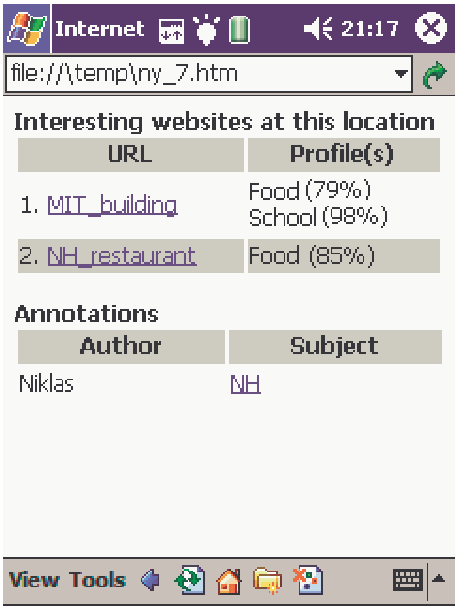

Figure 18 - The results of an URL request in the vicinity of the MIT building and the NH-restaurant

This shows how the IR is supposed to work; however, it is seldom possible to get as good relevancy scores as in this example as long as the relevancy server returns random values. As we can see the MIT-building is assigned with two scores, 79% for food (there is a cafeteria inside this building) and 98% for school. The NH restaurant is assigned with the score 85% for food. Any other buildings in the vicinity that have a connection to the user's profiles have been sorted out by the relevancy server since they did not meet the requirements of the threshold value that in this example was set to 80 out of 255 which is equivalent to 31% in relevancy score.

By clicking on the links associated with the locations the respective homepages are opened. This function is visualized in to the left in Figure 18 where the "NH_Restaurant" link has been clicked.

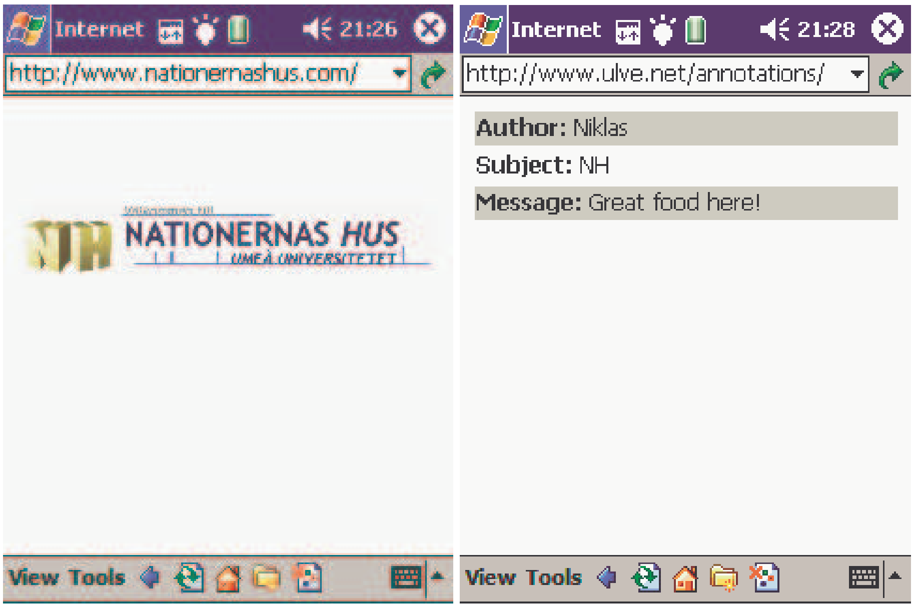

Figure 19 - To the left: Accessing the NH restaurant homepage from the IR result page. To the right: The annotation.

At the IR result page it is also possible to read the annotations placed on the virtual scribble board associated with the location. By clicking the subject link the annotation is displayed, this can be seen to the right in Figure 18. As we see someone by the name Niklas has placed a message here regarding the NH Restaurant stating that the food they serve here is good.

To perform further testing we try setting another annotation to the location. The annotation has an attached image file (Figure 19).

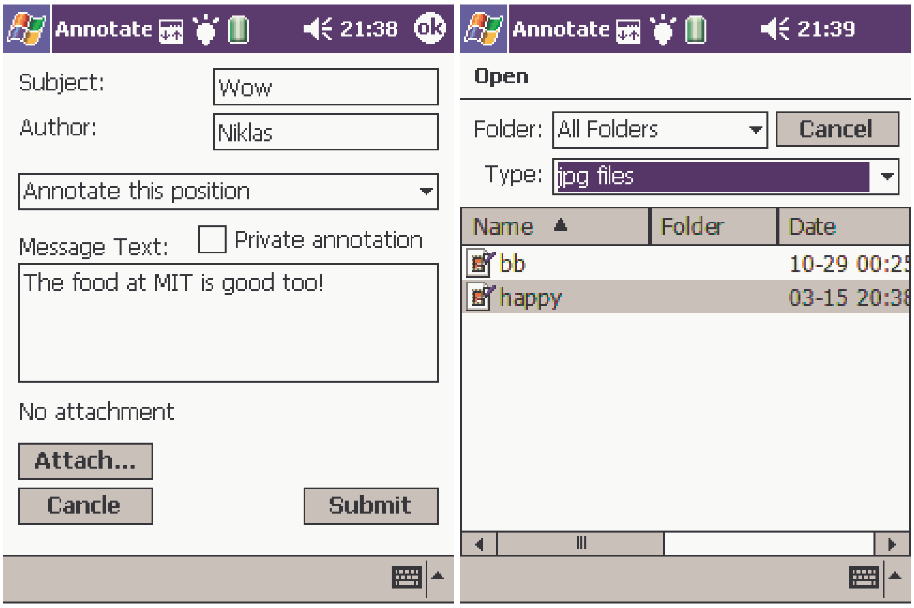

Figure 20 - To the left: Setting the annotation. To the right: Attaching a file.

To see the result of the annotation we do not move from the location. As the results of the new URL- and annotation requests arrive the IR notices that the new annotation has been added and we are alerted by a sound and the results are shown (Figure 20). Clicking the new annotation reveals what we expected, the message and the image shown in the web browser.

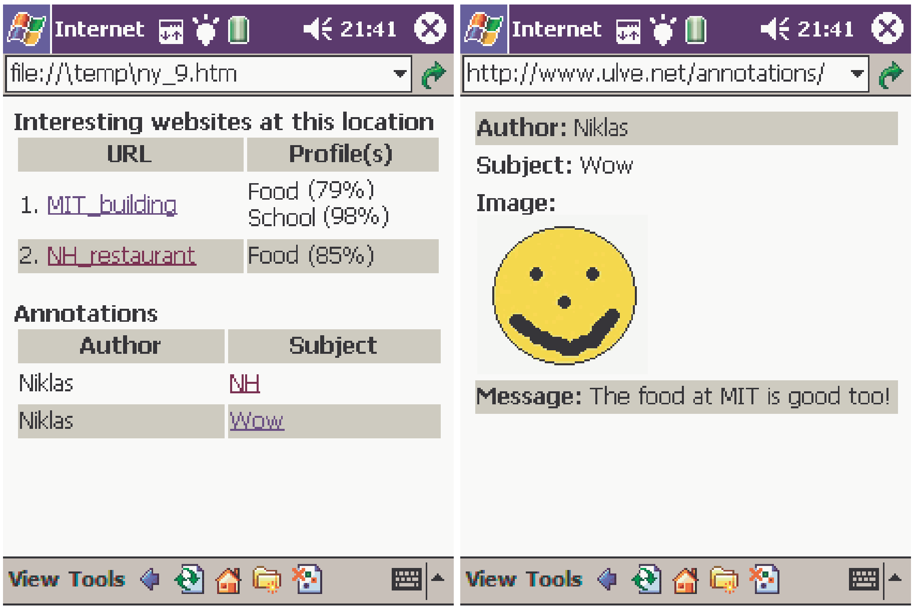

Figure 21 - To the left: The new annotation in the list. To the right: The new annotation is displayed.

While moving away from the location we are alerted by another beep. The NH restaurant has disappeared out of range of the 20 meter radius; however this is not what caused the beep. A new annotation has appeared among the results. Since another IR user Olov is within sight on the stairs of the MIT building and has an annotation attached to him that reads "Remind Olov about the thesis" we are alerted by the application as shown in Figure 21.

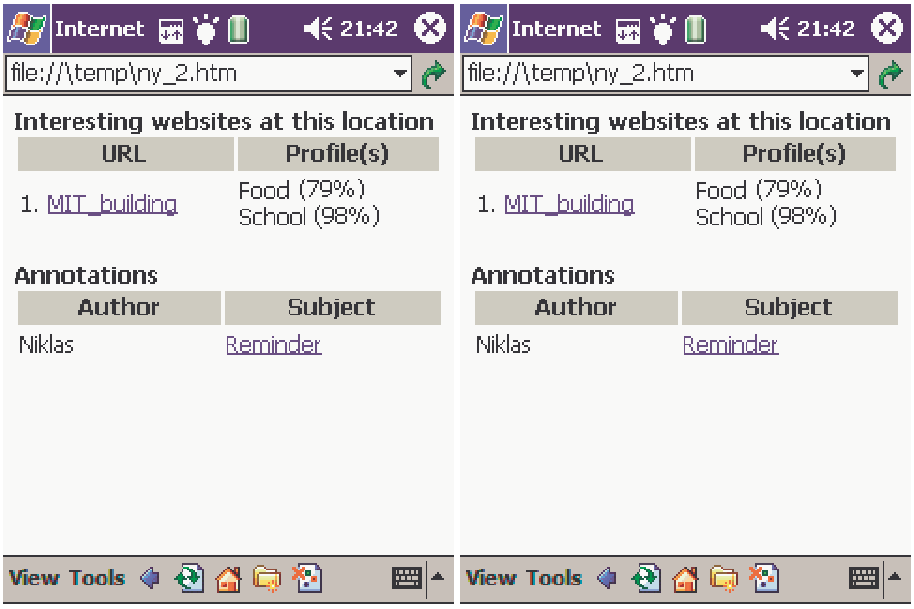

Figure 22 - An annotation attached to a person in the vicinity

4.5 Future Work

During the design process we have come across a multitude of ideas that we have wanted to incorporate in the finished product, some were not implemented because of various design choices, other ideas were deemed to require too much work to fit into the scope of this project and yet another category of ideas were regarded as visions for the future and left behind as inspirational musings.

Possible Extensions

All technical features that will and could be included in the application are not present in the prototype since time has not allowed it. While developing the prototype ideas have popped up regarding some ways in which the functionality and features of the program could be extended.

Haptic feedback: The current version of the prototype only supports sound notifications. The intention was to use the built in vibration function in the iPAQ 5550. Vibration strength should be directly associated with the relevance score of the received URL. A high score should result in strong vibration while a low score should give weak vibrations. At the time of development the HP support could not give any directions on how to access the vibration functions through programming the hardware. The only support found in the iPAQ SDK was for turning on the vibration function for calendar-notifications. This function did not help much since the vibration is at fixed strength and has limited usage.

History enhanced annotations: The affordance of digital artefacts can be highlighted by giving them history-recording functionality. This is for example evident in GeoNotes [Geo01] (see related work) where annotations can be sorted after popularity based on the actions of other users dealing (reading, saving or ignoring it) with the annotation. Functionality like this can aid in sorting out the most interesting annotations from the less interesting ones. As the amount of users' increase some places might get cluttered by annotations, this makes it important to offer good sorting possibilities.

Comments on annotations: Users should be able to make comments to annotations. This would increase the social aspect of the system since users can communicate in a chat-like manner by responding to others annotations and comments. This could also provide a good way of sorting out less interesting annotations by giving the user the possibility to sort by the number of comments made to an annotation.

Grading annotations: Another way of extending the annotation functionality is to give users the ability to manually grade them on a scale, say, from one to five to help others sort out uninteresting ones.

Ignore list: It is important to make it possible for users to delete notifications on uninteresting information. The user should with a simple click at a button mark an annotation or webpage as uninteresting to add it to a personal ignore list. Items on the ignore list will never be shown again to the user unless she removes the item from the list. This could be done by making a new database table containing ids of the annotations and web pages that are not interesting to the user.

Annotation handling: There should be some way of manipulating or at least deleting annotations for the user that have made them. The annotations are at this moment virtually stuck on the virtual scribble board when they are entered into the IRD by the server. The only way to remove them from the system is to manipulate the SQL database by hand. The annotations can however be overwritten by the user if the user overwrites the material on the FTP server. This should not be possible unless the server permits it since missing annotations on the WWW server can wreck havoc for the users.

Calibrating the GPS: The IR client should offer some kind of GPS-calibration possibility to avoid incorrect positioning. Faulty position values that are entered into the IRD are impossible to correct. Therefore it should be possible to calibrate the GPS to a fixed point, for example, the home (or any other fixed point with known location) of the user is always at the same location every day. By using the home location as a fixed point with known latitude and longitude it can be used to calculate an offset value for the latitude and longitude of the GPS receiver.

GeoURL meta tags: All annotations carry a meta-tag compatible with the GeoURL [Sch04] (see related work) system. This makes it possible to add all annotations to the GeoURL database. However, it is important to keep in mind that if annotations are deleted they must also be deleted from the GeoURL database to prevent it from being filled with dead links.

Visions

This system, or rather this type of system could very well be extended quite a bit in the future. This differs from the future work in that these extensions require a significant amount of extra work and possibly even complete redesign of the existing infrastructure, database and protocols making the only real remaining component the basic idea of connecting the virtual and the physical world and presenting a selection of the virtual world based on the user's physical location.

Smaller devices

The components of this project are today both a PDA more or less dedicated to running a special Information Radar Software and a GPS device that provides the PDA with positioning information. This approach is not feasible for mass deployment since the cost of both a PDA and a GPS is quite high and the number of uses is somewhat restricted for ordinary users. A second drawback with the PDA approach is the size of the PDA. It might not seem like a big device but it exceeds today's mobile phones both in volume and in mass making it rather unattractive to carry especially if its sole purpose would be the IR Software.

The price issue could be helped by a more ubiquitous approach. The need for a separate GPS receiver for every user could be handled by the sharing of GPS receivers between collocated users or by the use of smart spaces, transmitters built in to the infrastructure of houses and entire cities that among other things transmits approximate positions that could be used by these simplistic devices. A more down to earth approach would be to embed a GPS receiver in one of the user's devices for example in a mobile phone and that receiver could be shared among all other peripherals like cameras (annotating images with position), PDAs (for the IR Software or other position aware software) or the laptop. The mobile phone network could also be used for a cruder positioning and this is already available in several places. The mobile phone could make its position readable by other devices belonging to that person. Another example of an alternate approach would be to use WLAN positioning where available. Wireless network cards are currently more common among PDAs and laptops then GPS receivers and indoor positioning can be an attractive solution for either more restricted environments like a guide software in a museum or as an extension of the GPS that enables transparency between indoor and outdoor positioning.

The size of the device could be reduced but at a cost of the richness of the interface. If the positioning is done by a mobile phone or by ubiquitous means the size of the device could be significantly reduced, pen size or smaller would be possible. Interfacing with pen sized devices could however be a problem. A small LCD could be installed that could display short messages, haptic feedback could be used to alert the user that something interesting is close at hand or even auditory feedback could be used. Auditory and haptic modalities can be modulated in both frequency and amplitude to provide some way of differentiating between different notification strengths in a seamless manner (higher pitch for more important notifications etc). There exist other techniques for auditory feedback that would be suitable for devices with a minimalistic interface for example Earcons and Sonification.

Richer interfaces

In the future there might evolve a significantly richer interface then the current project provides as opposed to the minimalistic branch previously mentioned. There might in fact be a whole line of different manifestations of the core concept for different needs all using the same server infrastructure. While the ordinary user might suffice with a minimalistic device for basic notification needs more advanced users might want more functionality. The situation may also demand a richer interface a guide application might show additional information of the tour objects or the backpacker version may need some method of inputting data into the database for other users to read. Versions supporting laptops will permit use of a larger screen area making it practical with more information displayed simultaneously. The keyboard would also permit more lengthy text input. The significantly higher processor capacity will also allow more advanced manipulation of data both pre-processing before displaying it to the user but also before inserting annotation material into the database.

Extended functionality

Several ongoing projects might possibly be merged into the same framework as this project. This would enable the users to use the same interface to do several position-based tasks.

The IR system could be extended to report critical information to the user, information that is not necessarily interesting to the user in the form of personal likings such as football or fishing. A new database for "general information" could be created to store important location-specific information. For example, in case of a traffic accident the database could be updated with the location of the accident together with a message that warns the nearby users of the location and reports that the road is blocked.

To bring more social aspects into the system instant messaging (ICQ [icq96], MSN [msn97], Jabber [jab99] etc.) functionality could be added. For example, a contact list and possibilities for messaging and file transfers. This opens up a wide range of possible position based extensions that could be made. For instance, users could allow buddies to see their current physical locations and a user could send messages to many users that are present at a certain location.

4.6 Summary

When we set out to implement the IRS we had the goal of creating an application that could lighten the cognitive burden for the user of the client by actively keep a lookout for information in the vicinity of the user. In background of this the information would be websites with a physical connection. The information reported back to the user should also be such information that would be deemed interesting by the user and consequently the application needed to be context aware. At a later stage of the project the benefits of annotation were discovered and incorporating features for a virtual scribble board was added to the list of goals.

In light of this we developed several client-server systems but finally ended up with the rather complex system described in this chapter. The key points of the system were the two servers (the relevancy server and the IRS) and the IR client. The relevancy server were initially not part of this thesis project so we developed a dummy server that only returned random relevancy values to the enquiring client, the IRS handled most of the other server side functionality in particular communication with the IRD. The client was developed for a PDA and used a GPS module for positioning and a mobile phone for communication with the servers both of which communicated with the client via Bluetooth. Other parts of the system that were not implemented by us are the FTP server needed to store annotations, the WWW server for making annotations and information available to the clients a SQL database for storing information regarding users, annotations, URLs etc. Communication protocols were also designed for communication between the client and the servers.

If we were able to do the project once again with the knowledge that we have gained from this project some things would obviously have been done differently. First of all we would not have had to develop as many prototypes as we did this time around and we would have known what server layout that proved most useful. This is of course hard to do the first time around seeing as nothing was really known when we begun this project. Another priority would be to change the client-server protocol somewhat to optimize it some more. This would also require some change in the servers' layout but it would probably reduce the amount of communication radically. Also some server-server communication would be adapted so that the servers could cooperate for a more effective result.

In conclusion we can say that the technical aspects of this thesis project have succeeded. The goals set initially have all been met in some way or another. The context-aware portions are fulfilled by the use of the GPS for the positioning and also the profiles that enables relevancy checks for information. The test runs proved that the application indeed functions as the information radar that was the primary goal of this thesis; it was reporting relevant information to the user. The last goal, the capability for the user to annotate the environment or other users in the system also proved successful during the test run this meant that all three main goals were achieved and so the whole technical part of the project can be considered a success.

5. Conclusion

The focus in this thesis has been on the connection between information and location. From this point we have sought to develop a cognitive tool that reduces the problem of finding relevant elements in the growing space of information that increases both in availability and mobility. One of the aims of this thesis has been to examine and present the theoretical basis needed to understand what is important in construction of such a tool. This has resulted in two surveys, one reviewing the area of how we can make use of the environment and cognitive artefacts to improve our cognitive skills, and one that examines the areas of context awareness and mobile interfaces. As a practical aim we have also sought to develop a prototype that encompasses the basic features for finding relevant information by acquiring the context and location of the user. The aim has also been to include the possibility for users to add information to the world in the form of annotations connected to locations or other users, providing a functionality resembling a scribble board. The theoretical information gathered in the surveys has been of great inspiration for the ideas regarding the future and the development of the prototype.

From a cognitive perspective we have been inspired by the theories of how we make use of the environment in our everyday situations to aid our cognitive skills. As a cognitive artefact the IR changes the task of finding relevant information to listening for notifications. It also extends our knowledge of interesting locations by telling us about locations that we might not have been aware of before the IR notified us about them. We have also sought to prevent the IR from causing cognitive overload by minimizing the amount of interaction required between the user and the application once it has been started and configured. The virtual scribble board functionality of the IR can from a cognitive perspective be seen as an extension of the user's memory. It is possible to place information on locations and other users that work as reminders, once the user enters the vicinity of the location or person she is notified by the IR and the information is presented to her. In this way each user also has the ability to communicate with other users of the system by placing public annotations. The virtual scribble board can be seen as a social distribution of information among the users of the IR system.

The ideas discussed in the chapter on context awareness and mobile computing interfaces were only implemented to a relatively small extent in the final prototype. Many more things could have been done but they would in many occasions have taken too long time to implement and perfect, leaving the more fundamental functionality lacking. Some goals were achieved though, the device can notify the user by audio and the method implemented was the simplest possible method described in survey, auditory icons. The user can chose a sound that will represent a specific profile and thereby be able to interpret the meaning of an auditory notification. One other goal which was not achieved was the ability to use haptic feedback in order to minimize the disruption of nearby people.

The goals set for using context awareness to reduce the needed user interaction were met quite well. First of all, the information gathered from the GPS receiver was the base of the whole project and it turned out quite well in the end. The system will know where the user is all the time as long as line of sight can be drawn to the satellites in orbit around the planet. The information presented to the user is dependent on this information so the project would have been a total failure if this goal was not met. The second context aware functionality that was implemented was the profile management. The system does not only know who and where a user is it will also know what the user is interested in and only deliver information relevant to these three properties.

The aim of developing the IR prototype has not been to provide a system that incorporates all desired functionality that can be extracted from the ideas that we have presented in this thesis. Rather, the primary goal of this thesis has been to develop a basis for an application that can be improved in the future. As a next step in the line of development the IR prototype should be tested. User experiences and comments can then be used as a base for adding new functionality to the prototype. We consider the goal to be satisfactorily reached and it is our sincere hope that this prototype and the surveys presented within this thesis will give rise to more ideas and also encourage further development of the system.

6. Acknowledgements

Thanks to Anders Broberg for being our supervisor. Further thanks go to Anton Gustafsson for helping us with the various issues regarding the GPS receiver and to the computer science section at Umeå University for borrowing us hardware and literature.

7. References

[Abo00]: Abowd, G.D., Mynatt, E. D. (2000): *Charting past, present, and future research in ubiquitous computing*, ACM Transactions on Computer-Human Interaction pp. 29-58

[Alb98]: Albersm M.C, (1998):*The Varèse System, Hybrid Auditory Interfaces, and Satellite-Ground Control: Using Auditory Icons and Sonification in a Complex, Supervisory Control System* (citeseer.ist.psu.edu/204309.html), 2004-04-22

[Apa95]: *Apache HTTP Server Project* (1995): `http://httpd.apache.org/`, 2004-03-30

[Beg94]: Begault, D.R. (1994): *3-D sound for virtual reality and multimedia*, Academic Press Professional, Inc.

[Bjö99]: Björk, S., Bretan, I., Danielsson, R., Karlgren, J., Franzén. K. (1999): *WEST: A Web Browser for Small Terminals*, presented at UIST'99, Asheville, NC, 1999, pp. 187-196.

[Bjö00]: Björk, S., Redström, J., Ljungstrand, P., Holmquist, L.E. (2000): *POWERVIEW: Using information links and information views to navigate and visualize information on small displays.* In Proceedings of Handheld and Ubiquitous Computing 2000 (HUC2k), Bristol, U.K.

[Bla89]: Blattner, M., Sumikawa, D., Greenberg, G. (1989): *Earcons and icons: Their structure and common design principles.* Human Computer Interaction, 4(1), pp. 11-44

[Blu01]: *Bluetooth Special Interest Group* (2001): `http://www.bluetooth.com.`, 2004-02-02

[Bos03]: Bosman, S., Groenendaal, B., Findlater, J.W., Visser, T., de Graaf, M., Markopoulous, P. (2003): *GentleGuide: An exploration of haptic output for indoors pedestrian guidance.*

[Bre92]: Brewster, S.A., Wright, P.C., Edwards, A.D.N. (1992): *A detailed investigation into the effectiveness of earcons.*

[Bre93]: Brewster, S.A., Wright, P.C., Edwards, A.D.N. (1993): *An evaluation of earcons for use in auditory human-computer interfaces.*

[Bru01]: Brumitt, B., Shafer, S. (2001): *Better Living through Geometry, Personal Ubiquitous Computing*, Vol. 5, Number 1, pp. 42-45.

[Che00]: Cheverst, K., N. Davies, K. Mitchell and A. Friday, (2000): *Developing a Context-aware Electronic Tourist Guide: Some Issues and Experiences.* In Proc. of CHI'2000, Hague, Netherlands, 17-24, ACM Press.

[Che01]: Chevrest, K., Davies, N., Mitchell, K., Efstratiou, C. (2001): *Using Context as a Crystal Ball: Rewards and Pitfalls*, Personal Ubiquitous Computing, Vol. 5, Number 1, pp. 8-11.

[Ert98]: Ertan, S., Lee, C., Willets, A.,Tan, H. Z., Pentland, A., (1998*): A Wearable Haptic Navigation Guidance System*, Digest of the Second International Symposium on Wearable Computers, 164-165.

[Fre97]: Freundschuh, S.M., Egenhofer M.J, (1997): *Human conception of spaces*: Implications for Geofrapghic Information Systems. Transaction on GIS, p. 361-375

[Geo01]: *GeoNotes*, SICS, `http://geonotes.sics.se/`

[Ham95]: Hammond, K.J., Converse, T.M., Grass, J.W., (1995): *The stabilization of environments*. Artificial Intelligence 72: p. 305-327.

[Hei27]: Heidegger M., (1927): *Being and Time*

[Hut95a]: Hutchins, E., (1995): *Cognition in the Wild*. MIT Press.

[Hut95b]: Hutchins, E., (1995): *How a cockpit remembers its speed*. Cognitive Science, 19, p. 265-288.

[Icq96]: *ICQ*, ICQ Inc. (1996): `http://www.icq.com`, 2004-01-05

[Jab99]: *Jabber*: Jabber Software Foundation (1999): `http://www.jabber.org/` ,2004-03-01

[Joh97]: Johnston, M., Cohen, P.R., McGee, D., Oviatt, S.L., Pittman, J.A., Smith, I. (1997): *Unification-based Multimodal Integration*, Proceedings of the Thirty-Fifth Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics, Association for Computational Linguistics, Ed. Philip R. Cohen and Wolfgang Wahlster, pp. 281-288.

[Kir95a]: Kirsh, D., (1995): *The Intelligent Use of Space*. Artificial Intelligence, Vol. 73, Number 1-2, p. 31-68.

[Kir00]: Kirsh, D., (2000): *A Few Thoughts on Cognitive Overload*, Intellectica.

[Kir95b]: Kirsh, D., Maglio, P., (1995): *On Distinguishing Epistemic from Pragmatic Actions*. Cognitive Science.

[Kir99]: Kirsh, D., *Distributed Cognition, Coordination and Environment Design* (1999): Proceedings of the European conference on Cognitive Science p. 1-11.

[Lae01]: Van Laerhoven, K., Aidoo, K. (2001): *Teaching Context to Applications*, Personal Ubiquitous Computing, Vol. 5, Number 1, pp. 46-49.

[Mag99]: Maglio, P., Matlock, T., Raphaely, D., Chernicky, B., & Kirsh D., (1999): *Interactive skill in Scrabble*. In Proceedings of Twenty-first Annual Conference of the Cognitive Science Society.

[Msn97]: *MSN Messenger*, Microsoft (1997) `http://messenger.msn.com/`, 2004-04-21

[Nor91]: Norman, D.A., (1991): *Cognitive artifacts*, Designing interaction, Cambridge University Press, Cambridge.

[Mys95]: *MySQL AB (1995):* `http://www.mysql.com`, 2004-01-08

[Per99]: Perry, M. (1999): *The application of individually and socially distributed cognition in workplace studies: Two peas in a pod?* In Proceedings of the European Conference on Cognitive Science, p. 89-92.

[Pur01]: *PureFTPd (2001):* `http://www.pureftpd.org`, 2004-01-07

[Riz04]: Rizzo, A., Marti, P. (Unknown): *Distributed Cognition and Artifacts*, `http://www.sv.cict.fr/cotcos/pjs/TheoreticalApproaches/DistributedCog/DistCognitionpaperRizzo.htm`, 2004-02-24

[Rog97]: Rogers, Y., Scaife, M., (1997): *Distributed Cognition*, COGS, University of Sussex.

[Sca94]: Scaletti, C. (1994): *Sound synthesis algorithms for auditory data presentation*. In G. Kramer (Ed.), Auditory Display: Sonification, Audification, and Auditory Interfaces. Reading, MA: Addison-Wesley Publishing Company

[Scd94]: Schilit, B., Adams, N., Want, R. (1994): *Context-Aware Computing Applications*, Proceedings of the Workshop on Mobile Computing Systems and Applications.

[Sch04]: Schachter, J. (2004): *GeoUrl*, `http://www.geourl.org`

[Sci98]: Schmidt, A., Beigl, M., Gellersen, H-W. (1998): *There is more to Context than Location*, Proceedings of Workshop on Interactive Applications of Mobile Computing (IMC'98)

[Tar03]: Tarasewich, P. (2003): *Notification and Information Retrieval in Ubiquitous Environments*.

[Vit03]: *VITAL Projektet* (2003): `http://www.vital.cdt.luth.se/doc/home/see`, 2004-03-10

[Vyg74]: Vygotskij L.S., (1934/1974): *Thought and Language*. US, Cambridge: Cambridge, MIT.

[Wil02]: Wilson, M. (2002): *Six views of embodied cognition*. Psychonomic Bulletin and Review, 9, p. 625-636.

[Zha90]: Zhang J., (1990): *The Interaction of Internal and External Information in a Problem-Solving Task*, In Proceedings of the 13th Annual Conference of the Cognitive Science Society, p. 954-958

[Zie00]: Ziemke, T., (2001): *Are Robots Embodied?*. In Proceedings of the First International Workshop on Epigenetic Robotics: Modeling Cognitive Development in Robotic Systems. p. 75-93. Lund University Cognitive Studies, vol. 85.

Appendix A1: System Description - Client

The main loop

As soon as the GPS is started a timer starts to execute the functions in the main loop in order. Each function has to finish before the next function in order is started. All functions in the main loop are optional and it is up to the user to select which ones to run. The basic order of execution can be seen in figure 22.

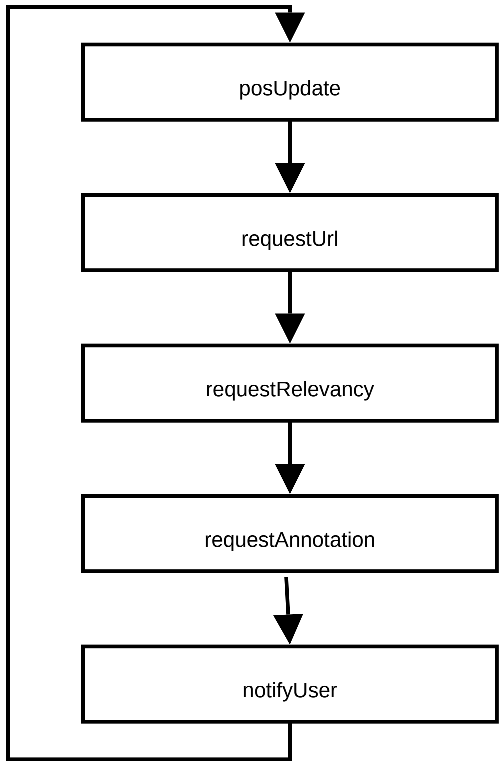

Figure 23 - The main loop

posUpdate: posUpdate updates the user's position on every loop, when this option is disabled the disabling function should do a UpdatePosition with coordinates -1, -1 to inform that the user no longer wants to be tracked.

requestUrl: requestUrl is in charge of requesting URLs from the server. If this function is disabled the requestRelevancy function should also be disabled since it no longer have any purpose.

requestAnnotation: requestAnnotation request annotations for the current area.

notifyUser: notifyUser should always be the last function called every cycle. It will notify the user by sound and/or a HTML summary.

posUpdate

The posUpdate function is in charge of reporting the user's current position to the server for tracking reasons. If the user should want to disable the tracking function the system should do a PositionUpdate with coordinates -1, -1 to indicate that the user no longer wants to be tracked. The order of execution and the basic dataflow can be seen in figure 23.

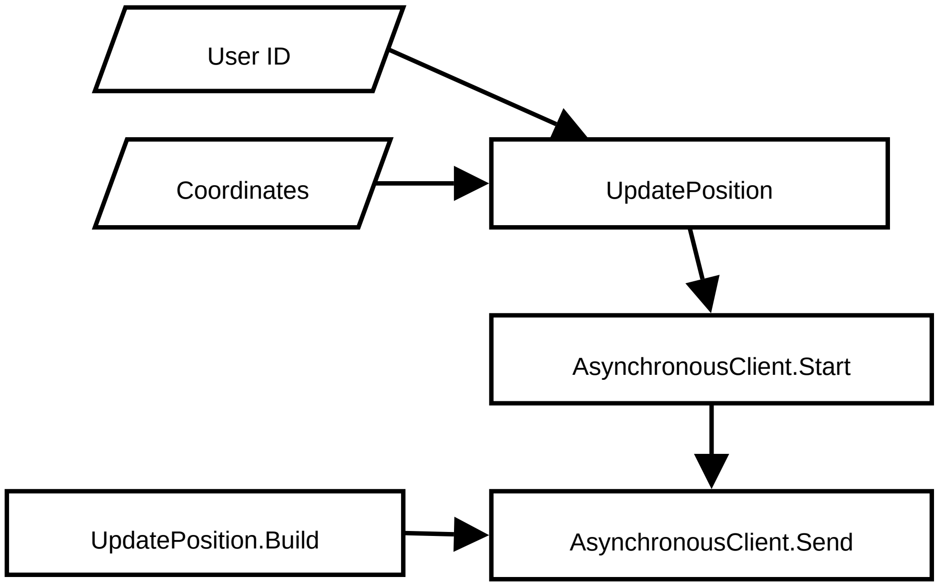

Figure 24 - posUpdate

User ID: The user ID assigned to the user by the system administrator.

Coordinates: Both latitude and longitude have to be entered as double values.

AsynchronousClient.Start: Starts a connection to the server.

UpdatePosition.Build: Builds the string that should be sent to the server.

AsynchrounousClient.Send: Sends a string to the server.

requestUrl

This function request nearby URLs from the server and stores them so that other functions can access them. The order of execution and the basic dataflow can be seen in figure 24.

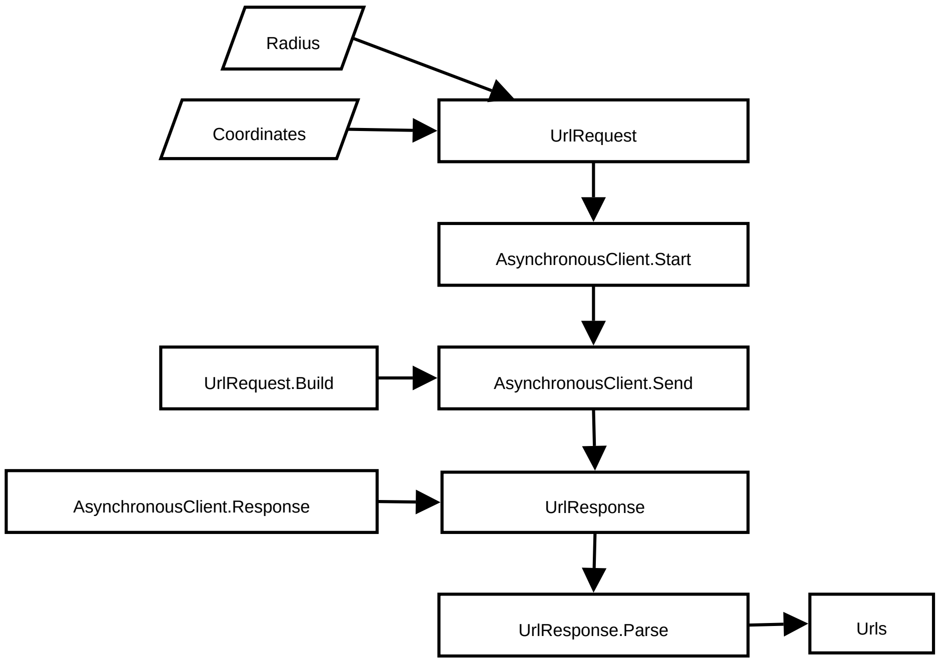

Figure 25 - requestUrl

Radius: The radius of the area centred at the user's current position. This is an integer value representing the radius in meters.

Coordinates: Both latitude and longitude have to be entered as double values.

AsynchronousClient.Start: Starts a connection to the server.

UrlRequest.Build: Builds the string that should be sent to the server.

AsynchrounousClient.Send: Sends a string to the server.

AsynchrounousClient.Response: A string containing the response from the server.

UrlResponse.Parse: Parses a string and stores the parsed values internally.

Urls: The global storage for recent URL values.

requestRelevancy

This function request relevancy information for the received URLs then stores the results so that they can be of use to other functions. The order of execution and the basic dataflow can be seen in figure 25.

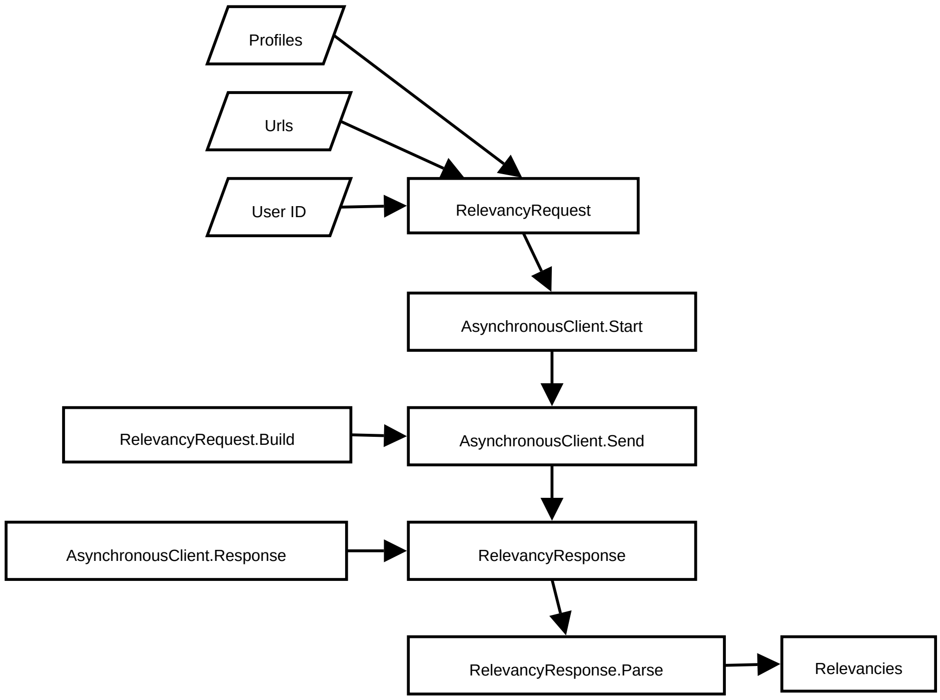

Figure 26 - requestRelevancy

Profiles: The profiles that relevancy will be requested for.

Urls: The URLs that relevancy will be requested for.

User ID: The user ID assigned to the user by the system administrator.

AsynchronousClient.Start: Starts a connection to the server.

UrlRequest.Build: Builds the string that should be sent to the server.

AsynchrounousClient.Send: Sends a string to the server.

AsynchrounousClient.Response: A string containing the response from the server.

RelevancyResponse.Parse: Parses a string and stores the parsed values internally.

Relevancies: The global storage for recent relevancy values.

requestAnnotations

This function will request annotations for the area specified and store the received annotations for use in other functions. The order of execution and the basic dataflow can be seen in figure 26.

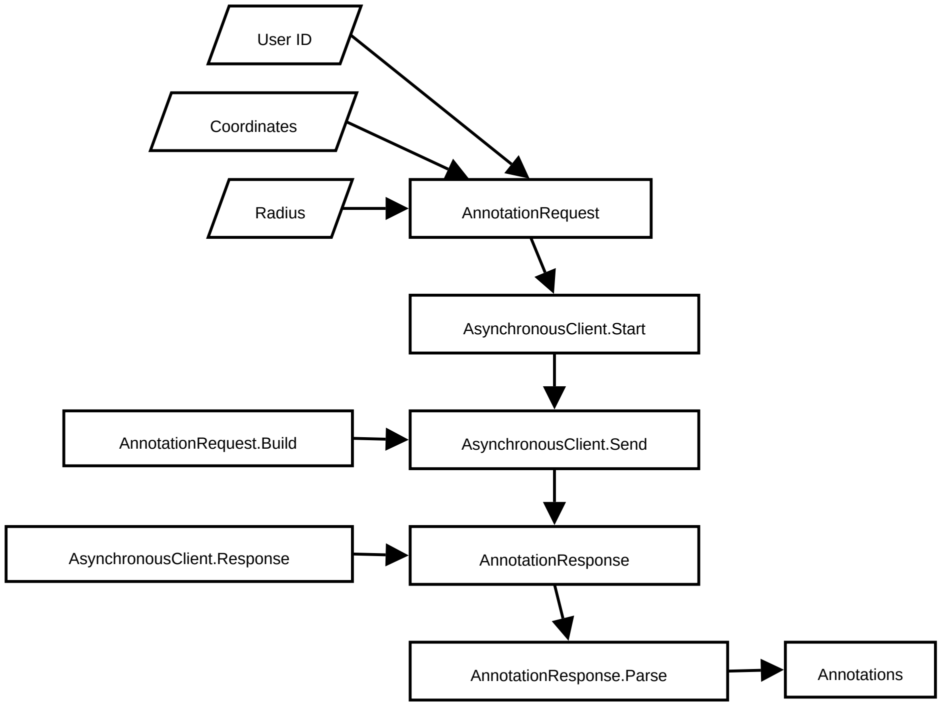

Figure 27 - requestAnnotations

User ID: The user ID assigned to the user by the system administrator.

Radius: The radius of the area centred at the user's current position. This is an integer value representing the radius in meters.

Coordinates: Both latitude and longitude have to be entered as double values.

AsynchronousClient.Start: Starts a connection to the server.

AnnotationRequest.Build: Builds the string that should be sent to the server.

AsynchrounousClient.Send: Sends a string to the server.

AsynchrounousClient.Response: A string containing the response from the server.

AnnotationResponse.Parse: Parses a string and stores the parsed values internally.

Annotations: The global storage for the most current annotations.

notifyUser

This function will notify the user if new annotations or URLs have arrived since the last function call. The order of execution and the basic dataflow can be seen in figure 27.

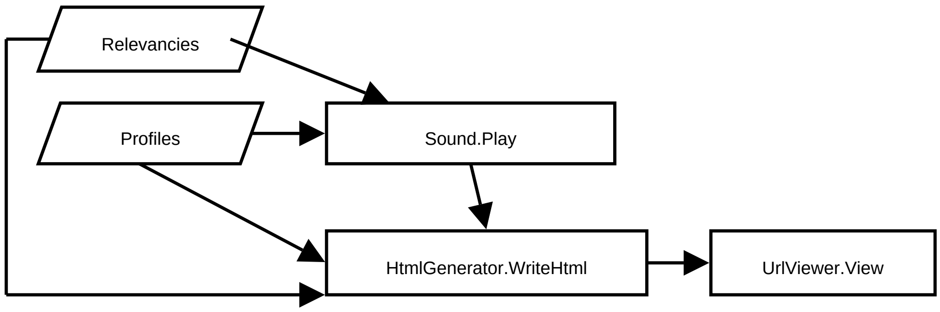

Figure 28 - notifyUser

Sound.Play: If the user have indicated that sounds should be played the notifyUser function plays the sound from the most dominant profile in Relevancies data structure. The appropriate sound is found in the Profiles data structure.

HtmlGenerator.WriteHtml: If the user have indicated that a HTML summary should be displayed the notifyUser function calls the HtmlGenerator.WriteHtml to produce such a HTML summary.

UrlViewer.View: If the user have indicated that a HTML summary should be displayed the notifyUser function calls the UrlViewer.View function to display the HTML page.

Annotate

When the user submits an annotation the system generates a HTML page which is uploaded to an FTP server then the Information Radar server is contacted and the new annotation is reported. The order of execution and the basic dataflow can be seen in figure 28.

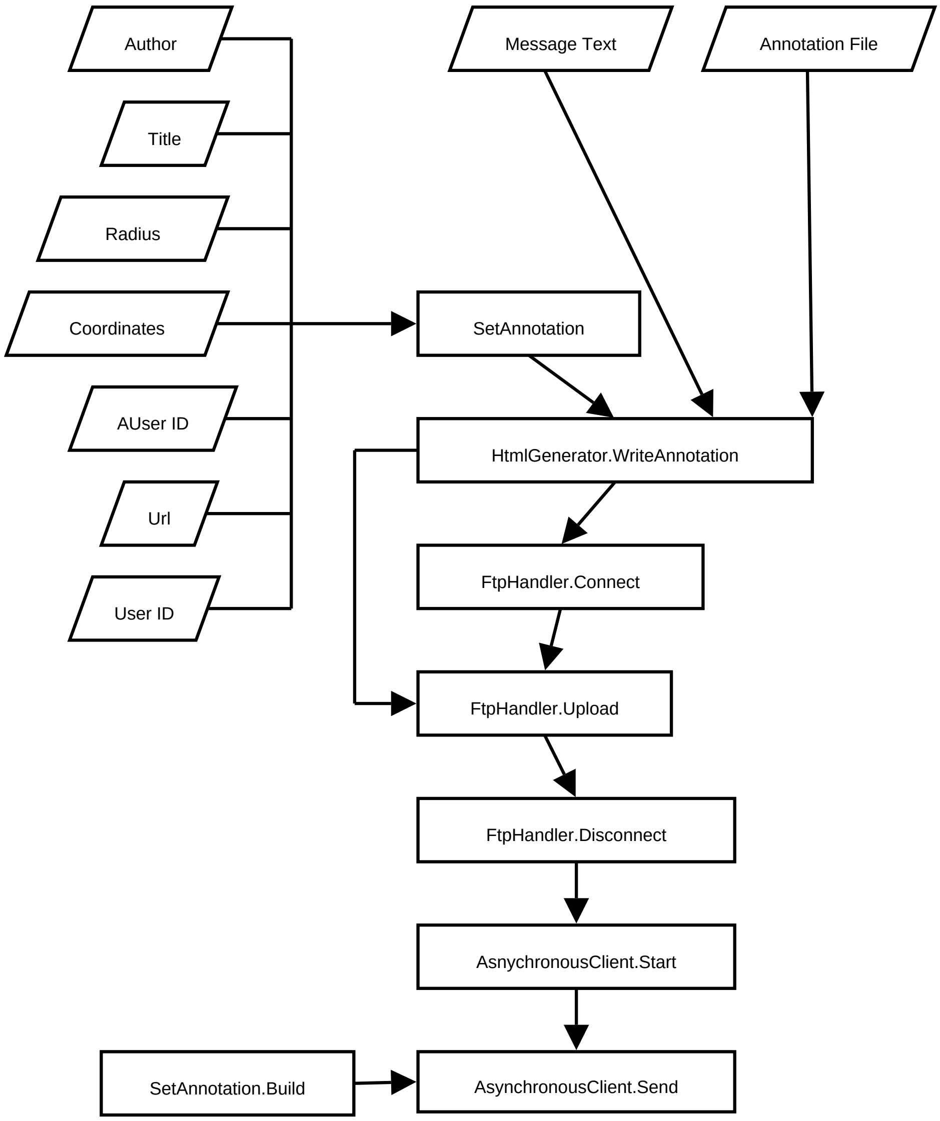

Figure 29 - Annotate

Author: A string containing alphanumerical characters. This string should be the author's name.

Title: A string containing alphanumerical characters. This string should be the title of the annotation.

Radius: The radius of the area centred at the user's current position. This is an integer value representing the radius in meters of the area where the annotation is valid.

Coordinates: Both latitude and longitude have to be entered as double values. If the annotation is private these values are automatically set to -1 -1.

AUser ID: This integer is set to the user id of the person that the annotation is attached to. If the annotation is attached to a location this is -1

Url: A string representing the URL to the annotation body.

User ID: The integer representing the user's id if this annotation is private, otherwise this is set to -1.

HtmlGenerator.WriteAnnotation: This function takes all relevant information regarding the annotation that is to be created and generates a HTML file in the temp directory.

FtpHandler.Connect: This function connects to the FTP-server.

FtpHandler.Upload: This function uploads a specified file overwriting any file already existing on the FTP server.

FtpHandler.Disconnect: This function disconnects from the FTP server.

AsynchronousClient.Start: Starts a connection to the server.

SetAnnotation.Build: Builds the string that should be sent to the server.

AsynchrounousClient.Send: Sends a string to the server.

ProfileRequest

This updates the profiles from the server. The order of execution and the basic dataflow can be seen in figure 29.

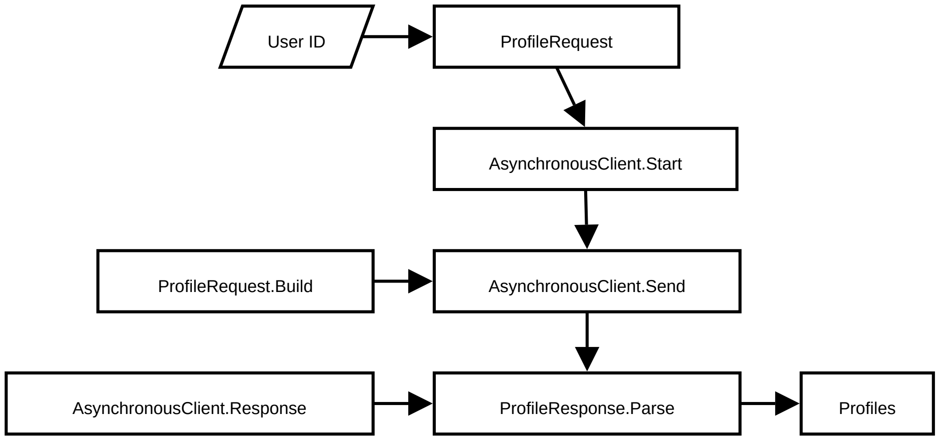

Figure 30 - ProfileRequest

User ID: The user ID assigned to the user by the system administrator.

AsynchronousClient.Start: Starts a connection to the server.

ProfileRequest.Build: Builds the string that should be sent to the server.

AsynchrounousClient.Send: Sends a string to the server.

AsynchrounousClient.Response: A string containing the response from the server.

ProfileResponse.Parse: Parses a string and stores the parsed values internally.

Profiles: The global storage for the most current profiles.

Appendix A2: System Description - Server

When the server is started it first parses the configuration file and then opens a socket that listens for client connections. As a client connects it accepts the connection and receives the data from the client. Then the data is parsed in the Parse-class and the command type read. After this the command specific variables are parsed. The variables are used to query the database and from the data returned by the database a response is formed and sent to the client.

There are some redundant variables and functions in the different command descriptions (see next page, figure 30 for an example of how it might look). These are covered here.

Async/RelevancyServer.Readcallback: Reads incoming data from the client.

CommandRequest.Parse: This method parses an incoming command request.

Parser.Parse: Parses incoming data from the client and examines which command type that has been sent.

Parser.DatabaseController.CommandName: Gets or sets the command specific data from the database.

Parser. CommandResponse: Holds the variables for making a command response.

Parser. CommandResponse.Build: Builds the command string that will be sent to the client.

Relevancy/AsyncServer.Send: Sends a response to the client

Note that CommandRequest equals the client requesting a command, for example, UrlRequest or AnnotationRequest. And CommandResponse is the respective response that is sent from the server to the client, for example, URLResponse or AnnotationResponse. The RelevancyServer denotes the main class for the relevancy server and the AsyncServer is the IRS.

UpdatePosition

This is how the IRS receives a update position command from the client. It updates the database and sends an acknowledge message to the client. The order of execution and the basic dataflow can be seen in figure 30.

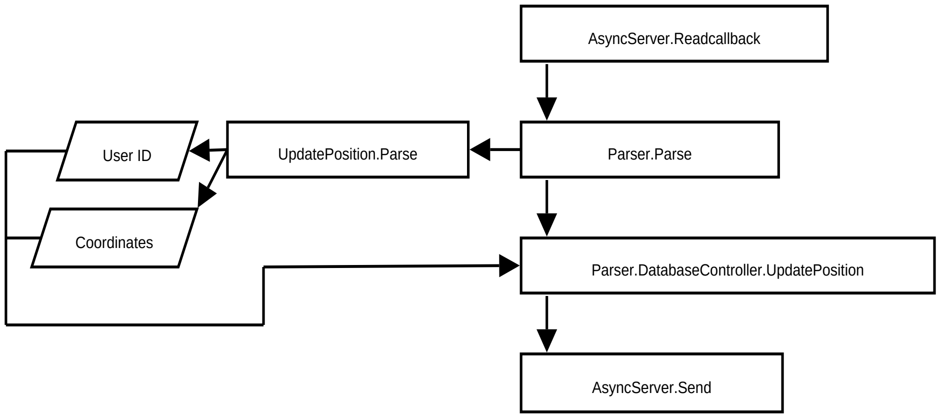

Figure 31 - UpdatePosition

User ID: The user ID integer value.

Coordinates: Latitude and longitude as double.

See protocol appendix for information regarding the response.

UrlResponse

This is how the IRS receives a URL request from the client and responds. The order of execution and the basic dataflow can be seen in figure 31.

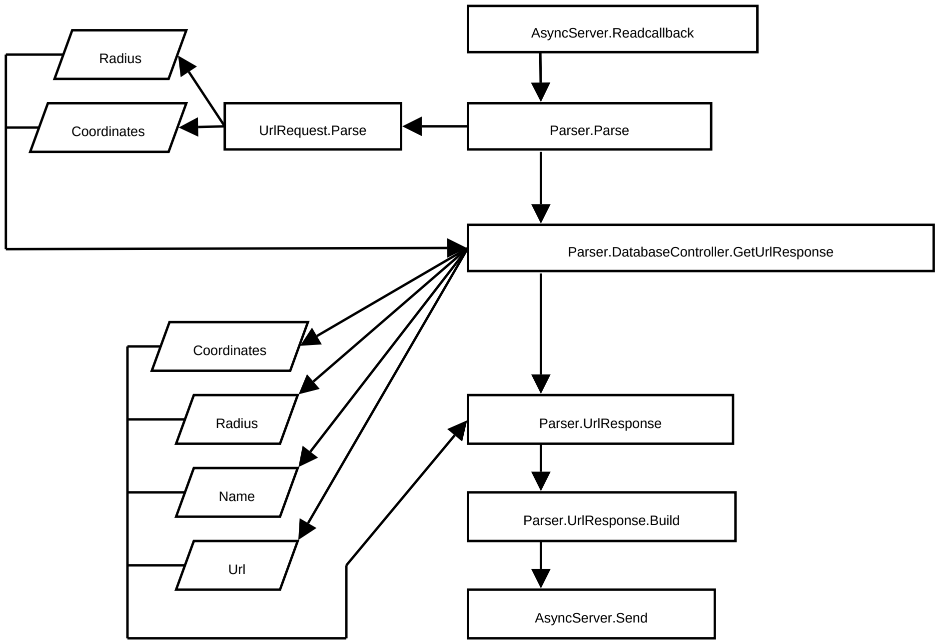

Figure 32 - UrlResponse

Radius: The radius of user. This is an integer value representing the radius in meters. In the URL response this value represents the radius of the URL, not the user's.

Coordinates: Latitude and longitude as double. In the response, these are the coordinates of the URL, not the user's position.

Name: Title string of the URL.

Url: The URL string.

The variables in the response are sent as a list so that many URLs can be sent within the same response.

RelevancyResponse

This is how the IRRS responds to a relevancy request sent from the client. The order of execution and the basic dataflow can be seen in figure 32.

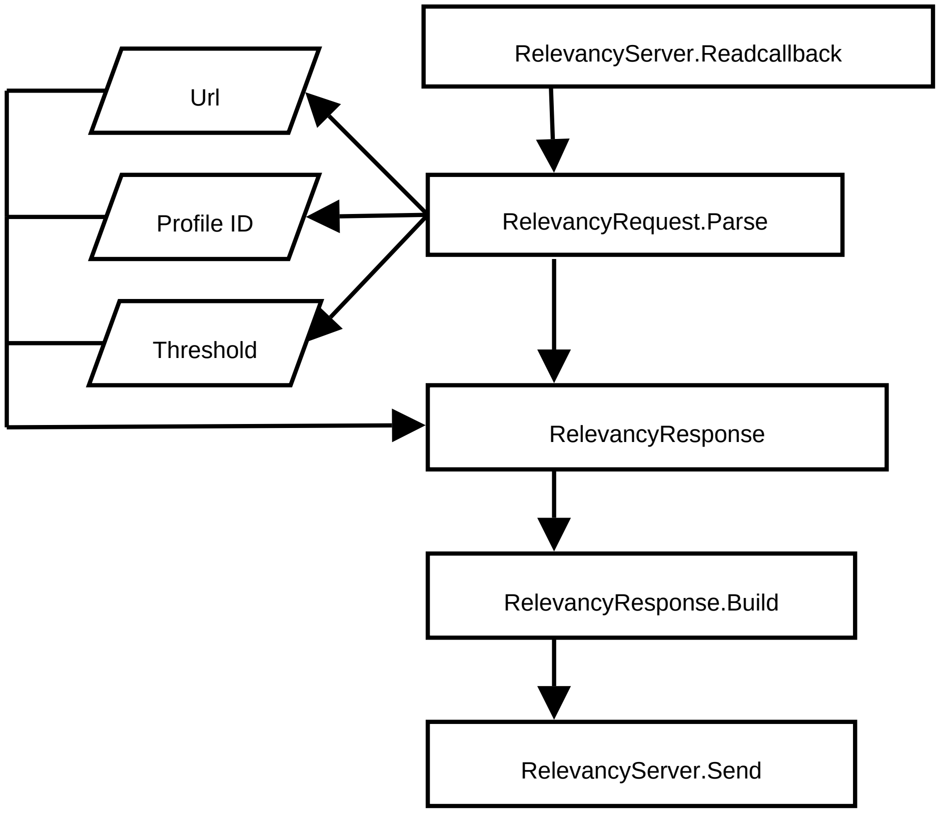

Figure 33 - RelevancyResponse

Url: List of URLs to be checked for relevancy.

Profile ID: The users profile ids.

Threshold: Threshold values for each profile.

Parser.RelevancyResponse.Build: Builds the string that will be sent to the client. All relevancies are randomized here.

See protocol appendix regarding details the response.

AnnotationResponse

This is how the IRS receives an annotation request from the client and responds. The order of execution and the basic dataflow can be seen in figure 33.

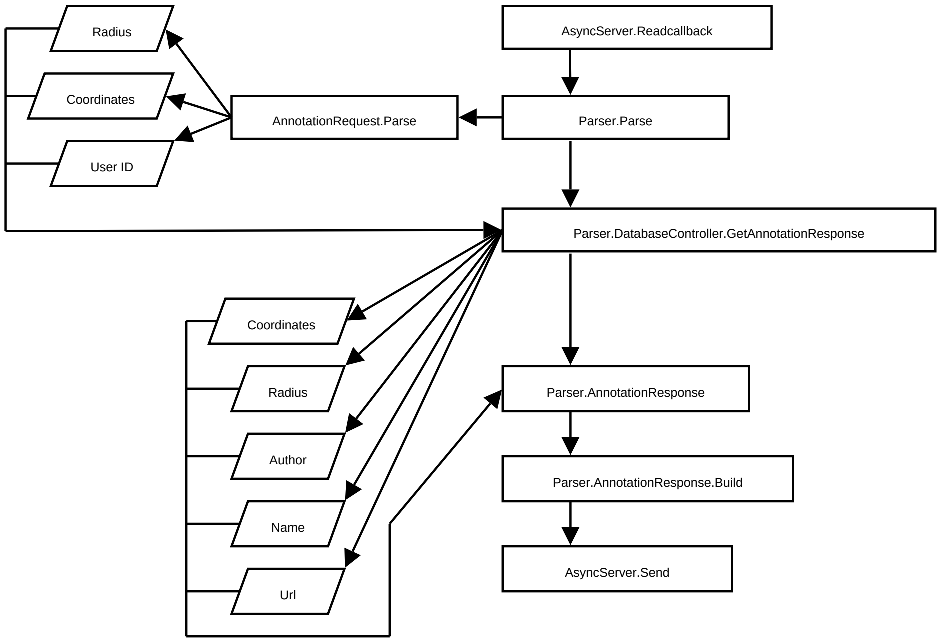

Figure 34 - AnnotationResponse

Radius: The radius of user. This is an integer value representing the radius in meters. In the annotation response this value represents the radius of the annotation, not the user's radius.

Coordinates: Latitude and longitude as double. In the response, these are the coordinates of the annotation, not the user's position.

Author: A string representing the author of the annotation.

Name: A string representing the title of the annotation.

Url: The URL to the annotation body.

The variables in the response are sent as a list so that many annotations can be sent within the same response.

Set annotation

This is how the IRS receives a set annotation command from the client. It updates the database and sends an acknowledge message to the client. The order of execution and the basic dataflow can be seen in figure 34.

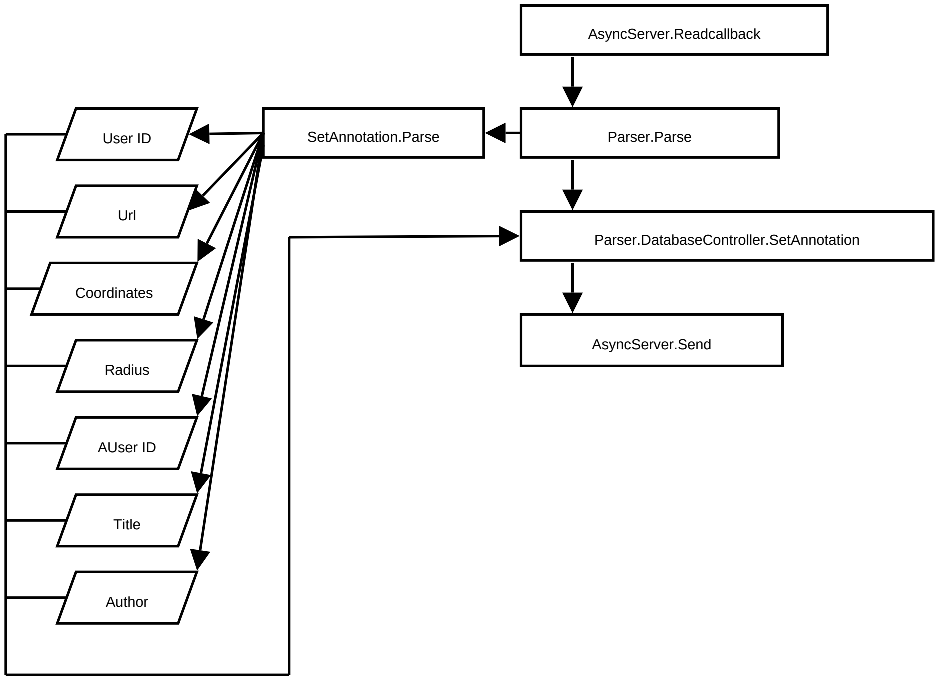

Figure 35 - SetAnnotation

User ID: The user's id or -1 if the annotation is public.

Url: The URL of the annotation body.

Coordinates: Latitude and longitude as double. These are set to -1 -1 if the annotation is connected to another person.

Radius: The radius integer of the annotation.

AUser ID: An integer showing if the annotation is attached to a user. If this is the case this variable will carry the user ID of the person, otherwise -1.

Title: A string representing the title of the annotation. This is the same variable as "Name" when requesting annotations.

Author: A string representing the author of the annotation.

See protocol appendix for information regarding the response.

User list response

This is how the IRS receives a user list request from the client and responds. The order of execution and the basic dataflow can be seen in figure 35.

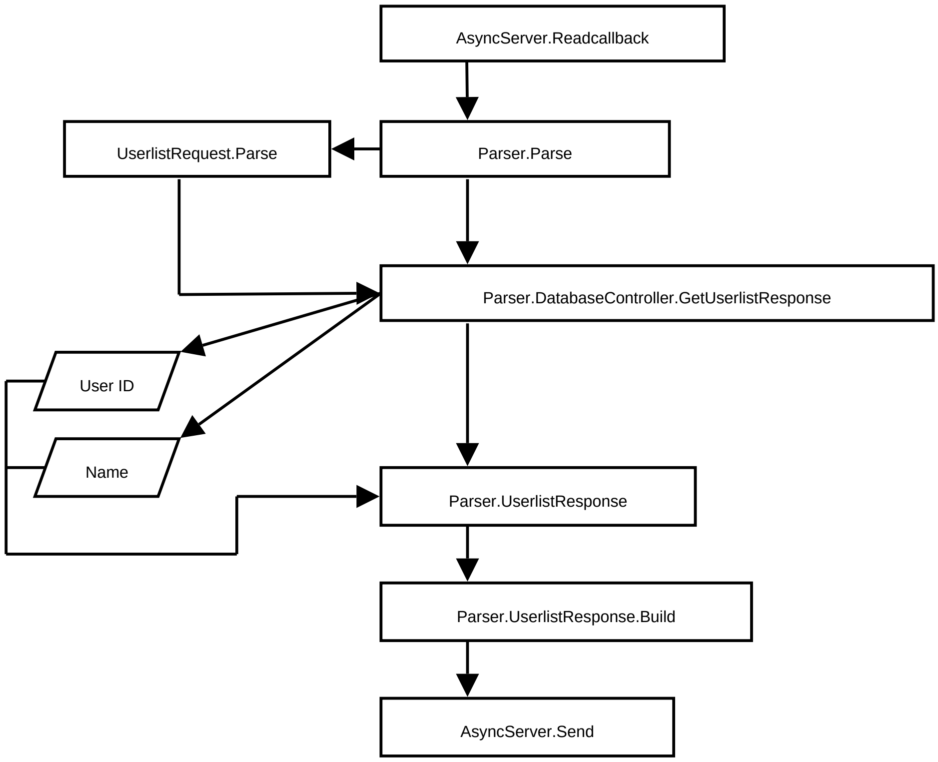

Figure 36 - UserListResponse

User ID: All user ID integers.

Name: All user name strings.

The variables in the response are sent as a list so that all users can be sent within the same response.

ProfileResponse

This is how the IRS receives a profile request from the client and responds. The order of execution and the basic dataflow can be seen in figure 36.

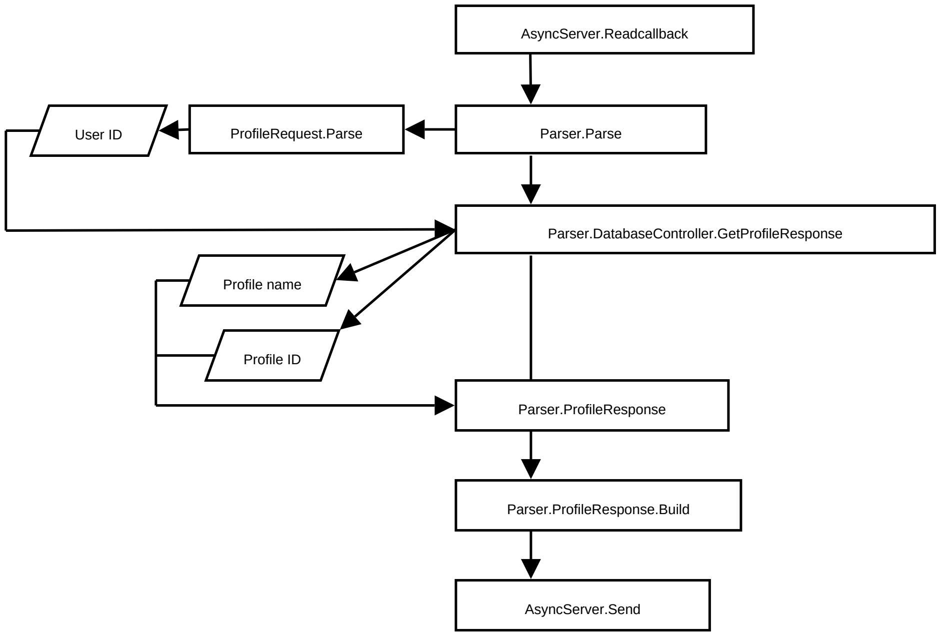

Figure 37 - ProfileResponse

User ID: The user's ID, integer.

Profile name: The users profile names.

The variables in the response are sent as a list so that many annotations can be sent within the same response.

Appendix B: Database Description

MySQL [mys95] was used to maintain the database. The server makes use of four different tables presented in Table 2.

USER				
id	name	latitude	longitude	categoryid
int(11) unsigned	varchar(50)	double	double	int(10)

Table 2 - The user table

URL				
latitude	longitude	radius	url	name
double	double	Int(11)	varchar(255)	varchar(30)

Table 3 - The URL table

PROFILE		
id	name	userid
int(11)	varchar(30)	int(11)

Table 4 - The profile table

ANNOTATION							
title	clientid	longitude	latitude	radius	url	author	isuserconnected
varchar(30)	int(11)	double	double	int(11)	varchar(255)	varchar(30)	int(11)

Table 5 - The annotation table

USER (table 2)

id: Key value. A unique integer to identify a user.

Name: User name in string format.

Latitude: A double value describing the location of a user. If the user is not online this will have the value of -1.

Longitude: A double value describing the location of a user. If the user is not online this will have the value of -1.

Categoryid: Not used yet, will be used in the future to categorize users.

URL (table 3)

Latitude: double value describing the location of the URL.

Longitude: Double value describing the location of the URL.

Radius: Describes the radius of the URL measured in meters.

URL: The URL

Name: A string describing the contents or title of the URL

PROFILE (table 4)

Id: Key value. A unique integer to identify a profile.

Name: A string describing the category or name of the profile.

Userid: An integer showing which user owns this profile.

ANNOTATION (table 5)

Title: A string describing the subject or title of the annotation.

Clientid: Denotes who owns the annotation, set to clientid if private, -1 if the annotation is public and readable for everyone.

Longitude: Double value describing the location of the annotation. If the annotation is attached to a user this is set to -1.

Latitude: Double value describing the location of the annotation. If the annotation is attached to a user this is set to -1.

Radius: Describes the radius of the annotation measured in meters.

url: The URL pointing to the annotation text.

author: The signature of the user.

isuserconnected: This is set to clientid of the user that the annotation is attached on. If no user is connected to the annotation this is set to -1.

Appendix C: Protocol Description

AnnotationRequest (client)

Description: This command is used to request annotations in a circular area, user ID of the requesting user, centre of the circle and the diameter of the circle is provided. The client initiates the communication with the server and this Command should result in an `ANNOTATION_REQUEST` being sent from the server to the client before the connection is broken.

Syntax: `ANNOTATION_REQUEST Latitude Longitude Radius UserID#`

All values are space separated, no space before the end of command marker #

Parameters: *Latitude* – A decimal value with decimal point representing the latitude of the centre of the circle where annotations will be returned.

Longitude – A decimal value with decimal point representing the longitude of the centre of the circle where annotations will be returned.

Radius – An integer value representing the diameter in meters of the circle where annotations will be returned.

UserID – An integer value representing the users ID number.

Example: `ANNOTATION_REQUEST 65.12982 47.192323 20 9#`

AnnotationResponse (server)

Description: This command is used by the server to send a list of annotations back to the client. The client initiates the communication and sends an `ANNOTATION_REQUEST` the server answers with `ANNOTATION_RESPONSE`.

Syntax: `ANNOTATION_RESPONSE Name Url Author Latitude Longitude Radius (Name Url Author Latitude Longitude Radius …)#` The string within the parenthesis could be repeated a arbitrary number of times. All values are space separated, no space before the end of command marker #

Parameters: *Name* – A string representation of the title of the url, this cannot contain any spaces.

Url – A string representation of the address to the annotation.

Author – A string representation of the name of the author, this cannot contain any spaces.

Latitude – A decimal value with decimal point representing the latitude of the centre of the URL's physical location.

Longitude – A decimal value with decimal point representing the longitude of the centre of the URL's physical location.

Radius – An integer value representing the diameter in meters of the circle that corresponds to the physical size of the URL.

Example: `ANNOTATION_RESPONSE MyAnnotation http://www.cs.umu.se/~masterthesis/test.html Olov 73.23123 12.121234 20 MySecondAnnotation http://www.cs.umu.se/~masterthesis/test2.html Niklas 84.34123 83.2131 10#`

ProfileRequest (client)

Description: The client initiates the connection to the server and sends this command then waits for a PROFILE_RESPONSE which contains a list of which profiles the user are a member of.

Syntax: `PROFILE_REQUEST UserID#` No space before the end of command marker #

Parameters: *UserID* – An integer value representing the users ID number.

Example: `PROFILE_REQUEST 2#`

ProfileResponse (server)

Description: This command is sent as a response to the `PROFILE_REQUEST`. It should contain a list of profile names and profile numbers that the requesting user is presently member of.

Syntax: `PROFILE_RESPONSE Name ProfileID (Name ProfileID …)#` The string within the parenthesis could be repeated a arbitrary number of times. All values are space separated, no space before the end of command marker #

Parameters: *Name* – A string containing the name of the profile without any spaces.

ProfileID – An integer value representing the profiles ID number.

Example: `PROFILE_RESPONSE Tv 4 Cars 9 DairyProducts 14#`

SetAnnotation (client)

Description: This command is used by the client to annotate a physical location.

Syntax: `SET_ANNOTATION Url UserID Latitude Longitude Radius AnnotatedUser Title Author#` All values are space separated, no space before the end of command marker #

Parameters: *Url* – A string representation of the address to the annotation.

UserID – An integer value representing the users ID number.

Latitude – A decimal value with decimal point representing the latitude of the centre of the annotation's physical location.

Longitude – A decimal value with decimal point representing the longitude of the centre of the annotation's physical location.

Radius – An integer value representing the diameter in meters of the circle that corresponds to the physical size of the annotation.

AnnotatedUser – The ID number of the user that should be annotated, -1 if the annotation does not have a connection to a specific user.

Title – A string containing the title of the annotation without any spaces.

Author – A string containing the authors name without any spaces.

Example: `SET_ANNOTATION http://www.cs.umu.se/~mastertheseis/annotation.html 4 43.234 65.234 5 -1 InterestingPlace Olov#`

UpdatePosition (client)

Description: This command is used by the client to update its position in the database.

Syntax: `UPDATE_POSITION UserID Latitude Longitude#` All values are space separated, no space before the end of command marker #

Parameters: *UserID* – An integer value representing the users ID number.

Latitude – A decimal value with decimal point representing the latitude of the user's physical location. Should be set to -1 if the user logs off.

Longitude – A decimal value with decimal point representing the longitude of the user's physical location. Should be set to -1 if the user logs off.

Example: `UPDATE_POSITION 5 43.23423 78.3422#`

UrlRequest (client)

Description: This command is used by the client to request a list of URLs whit physical representations within a given area.

Syntax: `URL_REQUEST Latitude Longitude Radius#` All values are space separated, no space before the end of command marker #

Parameters: *Latitude* – A decimal value with decimal point representing the latitude of the centre of the circle where URLs will be returned.

Longitude – A decimal value with decimal point representing the longitude of the centre of the circle where URLs will be returned.

Radius – An integer value representing the diameter in meters of the circle where URLs will be returned.

Example: `URL_REQUEST 23.234234 32.87798 50#`

UrlResponse (server)

Description: This command is sent as a response to the `URL_REQUEST` command containing la list with information about the URLs in the given area.

Syntax: `URL_RESPONSE Name Url Latitude Longitude Radius (Name Url Latitude Longitude Radius …)#` The string within the parenthesis could be repeated a arbitrary number of times. All values are space separated, no space before the end of command marker #

Parameters: *Name* – A string representation of the title of the URL, this cannot contain any spaces.

Url – A string representation of the address to the webpage.

Latitude – A decimal value with decimal point representing the latitude of the centre of the URL's physical location.

Longitude – A decimal value with decimal point representing the longitude of the centre of the URL's physical location.

Radius – An integer value representing the diameter in meters of the circle that corresponds to the physical size of the URL.

Example: `URL_RESPONSE Arla http://www.arla.se 43.1356 56.6423 35 Norrmejerier http://www.norrmejerier.se 56.634356 32.7898 100#`

UserlistRequest (client)

Description: This command is sent by the client to request a list of users in the system.

Syntax: `USERLIST_REQUEST#` No parameters at all No space before the end of command marker #

Parameters: None.

Example: `USERLIST_REQUEST#`

UserlistResponse (server)

Description: This command sends a list of all the users in the system to the requesting user.

Syntax: `USERLIST_RESPONSE Name UserID (Name UserID …)#` The string within the parenthesis could be repeated a arbitrary number of times. All values are space separated, no space before the end of command marker #

Parameters: *Name* – A string containing the name of the user without any spaces.

UserID – An integer value representing the users ID number.

Example: `USERLIST_RESPONSE Olov 14 Niklas 33#`

RelevancyRequest (client)

Description: The user sends a list of URLs and a list of profiles to the server and should receive a response with each URL and the relevancy rating for each profile.

Syntax: `RELEVACNY_REQUEST Url (Url …) |ProfileID Threshold (ProfileID Threshold …) #` All values are space separated, the two lists are separated by | and there are a space before the end of list separator | and before the end of command marker #

Parameters: *Url* – A string representation of the address to the webpage.

ProfileID – An integer value representing the profiles ID-number.

Threshold – An integer value representing the minimum relevancy to be returned.

Example: `RELEVANCY_REQUEST http://www.arla.se http://www.cs.umu.se |9 240 9 55 12 150 #`

RelevancyResponse (server)

Description: The server's response to `RELEVANCY_REQUEST` containing a list of URLs each with a list of profiles and relevancy ratings.

Syntax: `RELEVANCY_RESPONSE Url ProfileID RelevanceRating (ProfileID RelevanceRating …) |(`or `#` if this is the last post in the list`) (Url ProfileID RelevanceRating (ProfileID RelevanceRating …) |…) #`

Parameters: *Url* – A string representation of the address to the webpage.

ProfileID – An integer value representing the profiles ID number.

RelevancyRating – An integer value representing the relevancy in respect to the previous profile.

Example: `RELEVANCY_RESPONSE http://www.aftonbladet.se 5 35 8 99 17 188 |http://www.norrmejerier.se 9 180 12 255 1 7 #`

Appendix D1: Manual - Client

Installation

Installation of the two clients is best done through the Setup.exe program. Follow the instructions until prompted which packets to install (figure 38) there the two packets of interest for the client user is XP Client or PPC Client, select one or both of those and proceed to select where to install. The PPC Client should automatically be installed on the PDA on the next sync occasion. The PPC client can then be started by selecting the appropriate program file in the programs folder. The Windows version of the client should be available from either the icon on the desktop or from the start menu when properly installed.

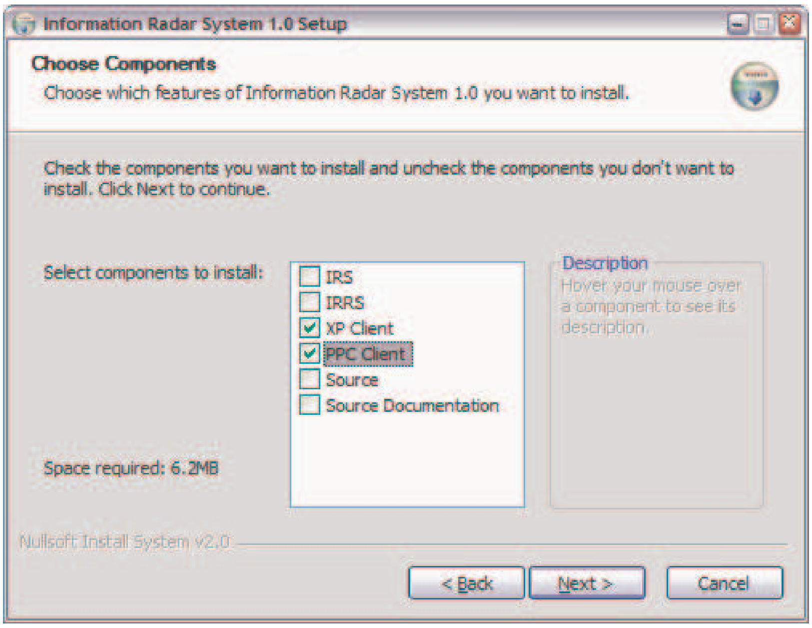

Figure 38 - Installation program

In order for the client to work Microsoft's .NET Compact Framework has to be installed this is acquired from their website at:

`http://msdn.microsoft.com/mobility/prodtechinfo/devtools/netcf/`

One other setting that not necessarily have to be done but that will greatly improve the use of the client is to turn of the function that enables the PocketPC device to turn itself off on inactivity. If this function is not disabled the device cannot be left to search for interesting information passively. To turn this function of go into the Settings menu on the PocketPC under systems and then under power uncheck the box that says "Turn off device if not used for" or consult the PocketPC manual.

Operation

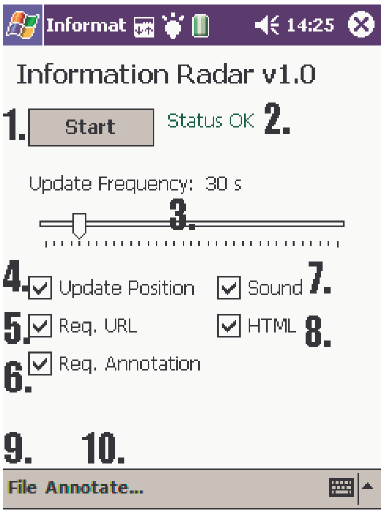

Figure 39 - Main screen

Main screen (figure 37)

1. Start button: When this button is pressed the program is going to ask for a Bluetooth GPS device that will supply the program with GPS data.

2. Status text: Status messages from the program will be displayed here.

3. Frequency slider: Used to set the frequency that the program should poll the different servers. This value can be anything between 10 seconds and 180 seconds.

4. Update pos: If this checkbox is checked the program will report its position to the server every time interval.

5. Req. URL: If this checkbox is checked the program will request nearby URLs every time interval. It will also request relevancies for the received URLs.

6. Req. Anno: If this checkbox is checked the program will request nearby annotations every time interval.

7. Sound: If this checkbox is checked the program will play a sample every time a new annotation or URL is received. The sound corresponds to the selected sound for the most dominant profile in the received relevancies.

8. HTML: If this checkbox is checked the program will generate and display a HTML summary of the nearby URLs and annotations.

9. File: The file menu where setting forms can be accessed and the program can be closed down.

10. Annotate: This menu option will open the annotation form and let the user annotate the current physical position with text and other attachments.

Main Screen Menus (figure 38)

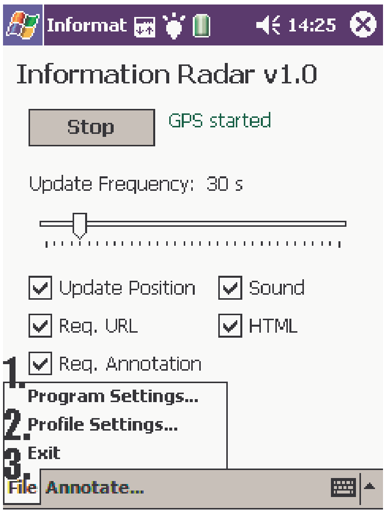

Figure 40 - File menu

1. Program Settings: This menu option will open the program settings form and set options regarding the operation of the program.

2. Profile Settings: This menu option will open the profile settings form and let the user set profile specific options.

3. Exit: This menu option will update the position to the offline position and exit the program.

Program settings (figure 39)

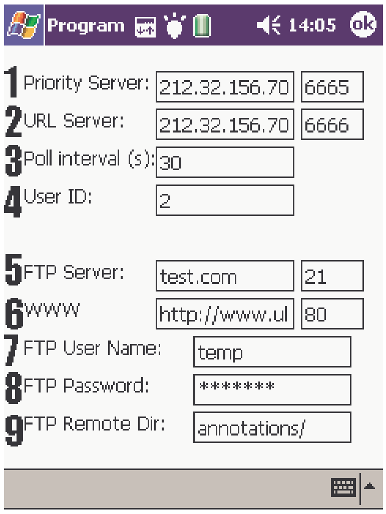

Figure 41 - Program settings

When this form opens up the previous configurations are read from file, anything can be changed and when the form is closed the new configurations are saved to file.

1. Rel. Server: The IP-address of the relevancy server the second text box is for the port number on which the server is running.

2. URL Server: The IP-address of the URL server the second text box is for the port number on which the server is running.

3. Poll interval: The interval on which the different servers should be queried.

4. User ID: The user ID-number assigned to the user by the system administrator.

5. FTP Server: The address to the FTP server where annotations should be uploaded. The second text box is for port number.

6. WWW Server: The address of the WWW server where the annotations can be viewed the second text box is for the port number.

7. FTP User Name: The username required to log in to the FTP server.

8. FTP Password: The password required to log in to the FTP server.

9. FTP Remote Dir: The directory where annotations should be uploaded.

Profile settings (figure 40, 41)

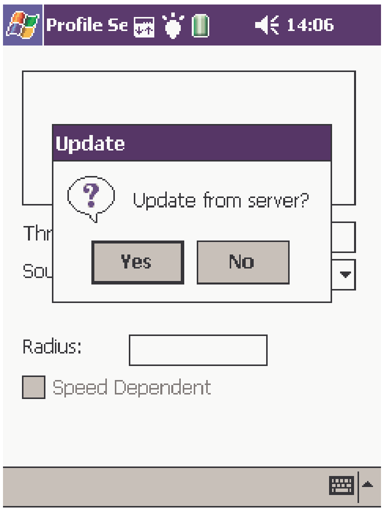

Figure 42 - Server update

When the profile settings form opens up the user is able to select if the profile settings are going to be updated from the server (figure 40). If settings are updated from the server all local settings will be lost.

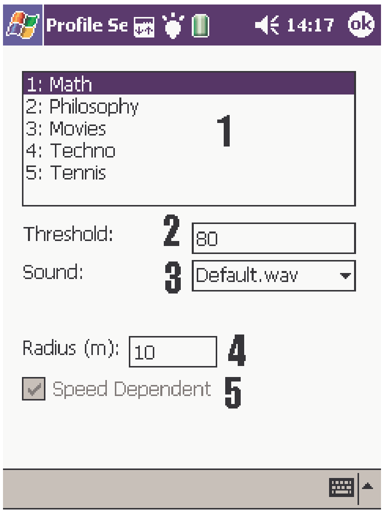

Figure 43 - Profile settings

When the profiles have been read either from disk or updated from the server all configurations regarding the profiles can be changed and saved to file (figure 41). When the user selects a profile in the profile list the threshold and sound updates to show the current settings for that particular profile. When the form is closed the settings are saved to file.

1. Profiles: A list of the user's current profiles. When a profile is selected the Threshold and sound changes to the values associated with that particular profile.

2. Threshold: The minimum relevancy number required to be returned from the relevancy server. This is an integer between 0 (return all) and 255. This value is bound to the selected value in the Profiles list box.

3. Sound: Which sample to play when this profile is the dominant one. This value is bound to the selected value in the Profiles list box.

4. Radius: The radius in metres where profiles and annotations should be returned. This value is used globally.

5. Speed Dependent: Automatically adjust the radius depending on speed and update interval so that no information sites are missed.

Annotate (figure 42)

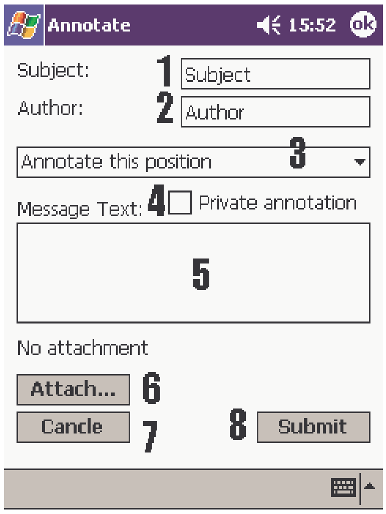

Figure 44 - Annotate

1. Subject: The subject of this annotation. This will be shown to other users on their summary screen it will also be used as the title of the generated web page. This line can only contain alphanumeric characters and no spaces.

2. Author: The name of the author, this will be shown to other users on their summary screen. This line can only contain alphanumeric characters and no spaces.

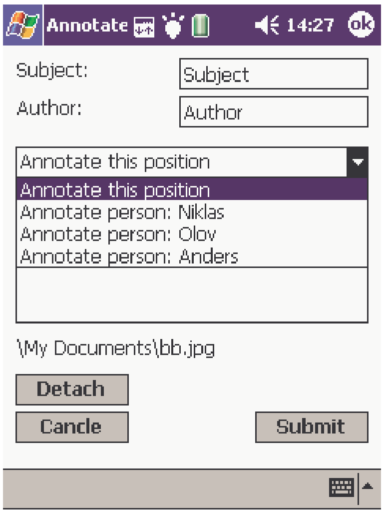

Figure 45 - Select which object to annotate

3. Annotation object: The object that should be annotated (figure 43). Defaults to the current physical position but can also be a certain user in the system.

4. Private annotation: If this check box is checked this annotation will only be visible to the user that made the annotation.

5. Message Text: The main body of the annotation. This line can only contain alphanumeric characters.

6. Attach: This button will bring display a file selector so that the user can select a file to be sent as annotation material.

7. Cancel: Cancels the annotation and returns to the main screen.

8. Submit: Sends the annotation to the FTP server and adds a link to the newly created web page on the URL server.

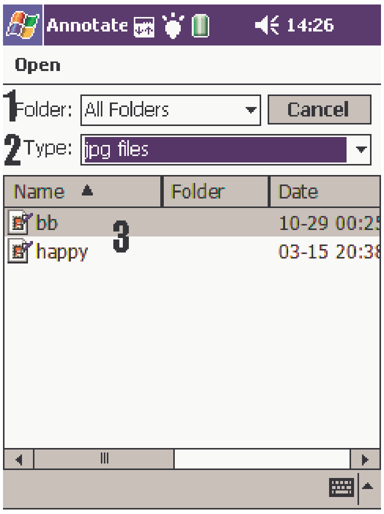

Figure 46 - Select annotation Material

When the user selects to annotate with a file the file selector opens up (figure 44)

1. Select Folder: This drop down box will let the user select which folder to look for annotation material in.

2. Select File type: This drop down box will let the user select which file types to select from.

3. Select File: Here the actual file that will be used as annotation material can be selected.

Summary Screen (figure 45)

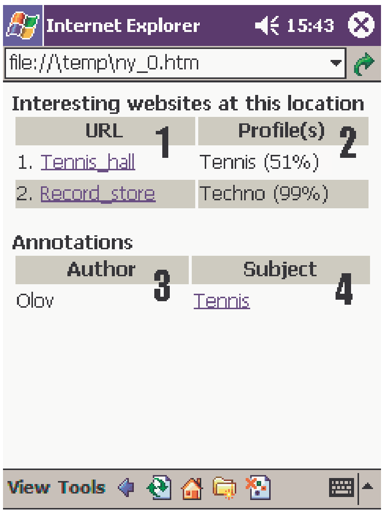

Figure 47 - Summary

On this screen a summary of the newest annotations and URLs will be shown. This is displayed through an external program, Pocket Internet Explorer. If the program is closed the user will be returned to the main program.

Websites

The first section is interesting websites, this is web sites close to the user's physical location that are deemed interesting by the relevancy server. Clicking a link will open a new web page.

1. URL: Here links to interesting sites will be shown. If the user clicks a link the browser will commence showing the selected site. The summary screen should be accessible by pressing the back button from the new site.

2. Profiles: Here the relevant profiles and the relevancy rating for each profile will be shown. One single URL can have several profiles associated with it.

Annotations

The second section contains annotations made by other users of the system. They may contain text, images, sounds or possibly other annotation material. There is no guarantee that the author names and topics are truthful entered. Clicking an annotation will bring a new web page displaying the material.

3. Author: Here the authors of annotations will be listed. Their names are chosen by the annotation user so there is no guarantee that they will be correct.

4. Annotation: A link to the annotation. If the user clicks a link the browser will commence showing the selected annotation. The summary screen should be accessible by pressing the back button from the new site.

Appendix D2: Manual - Server

The installation of the servers is quite straight forward. Execute the Setup.exe file and follow the instructions until prompted which parts of the system to install (figure 48). The options IRS and IRRS are the two options which install the different servers. If they are selected the user is prompted where to install the servers and where to place the start menu shortcuts.

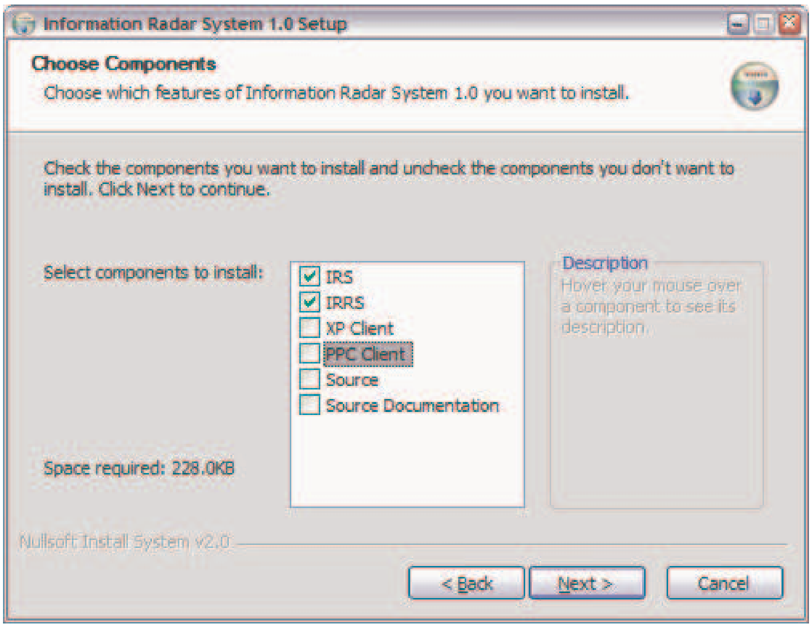

Figure 48 - The installation program

The server is started by double clicking the IRS.exe icon. It can also be started from the command prompt by typing "IRS.exe" when inside the server directory or selecting the program from the start menu. The IRRS is started in the same way, but here the filename is "IRRS.exe". See figure 46 and 47 for screenshots on the servers when executed. When a server is started it needs no further supervision as long as no fatal errors occur that might cause it to crash.

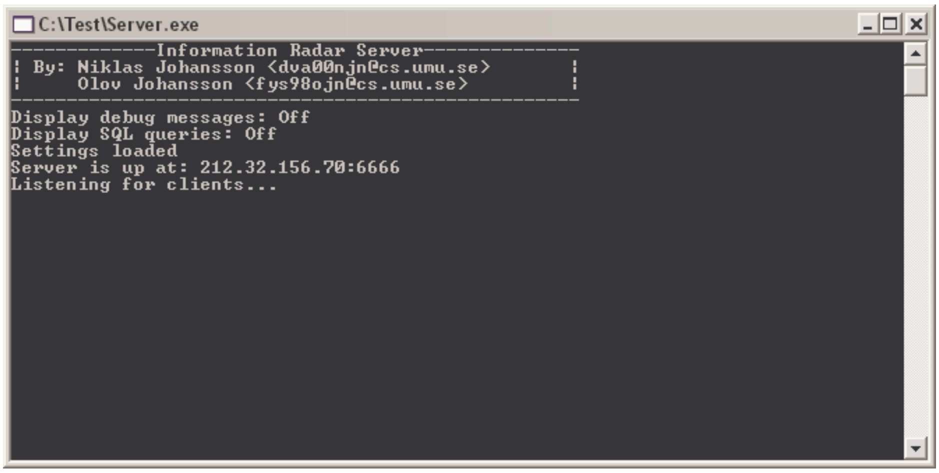

Figure 49 - The IRS when executed

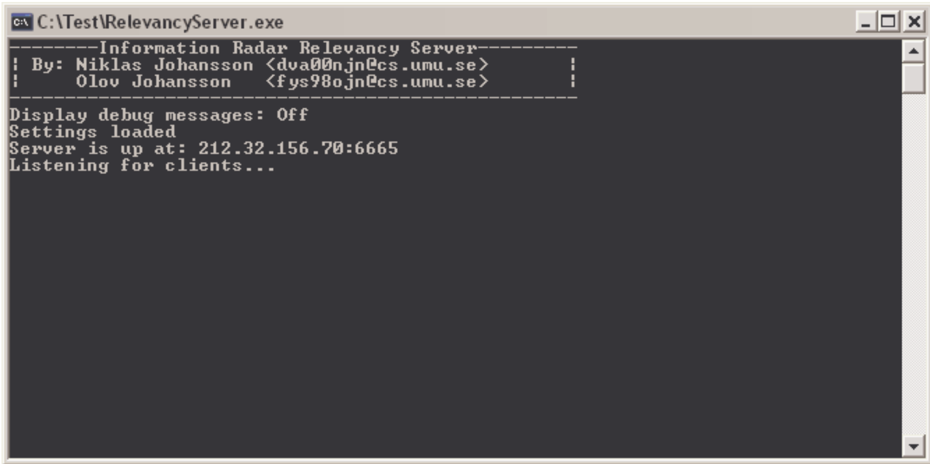

Figure 50 - The IRRS when executed

If the any of the servers does not start as expected or display an error message it could be wise to review the settings. This could be done by manually editing the settings file, this is however not recommended. A better way to do this is by using the settings applications that automatically update the configuration files.

To start the IRS settings application, double click the icon or type the filename in the command prompt (IRS Settings.exe). The same procedure holds for the IRRS settings program (IRRS Settings.exe).

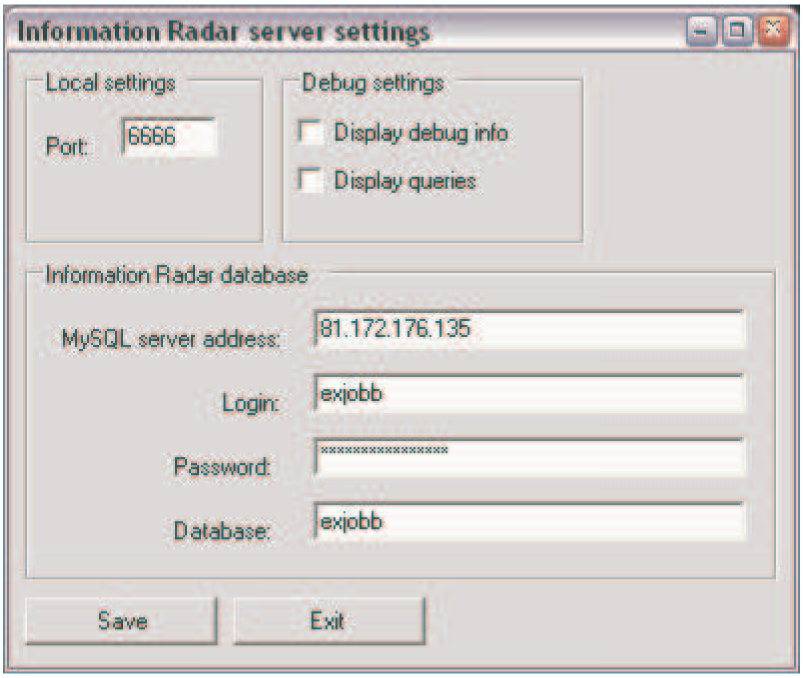

Figure 51 - The IRS settings application

The IRRS settings application is visualized in figure 48. A description of the various options follows:

Port: The port of the incoming requests and updates from the client. The default value for this is 6666. Any free port number could be used.

Display debug info: If this box is checked the server will display various debug information regarding client connections, parsed command messages etc.

Display queries: If this box is checked all queries sent to the Information Radar Database (IRD) will be printed.

MySQL server address: This is the IP-address of the IRD.

Login: The login name for the IRD.

Password: The password required to log in to the IRD.

Database: The name of the database that holds the tables needed by the IRS (see the database appendix for detailed information).

If none of the debug settings are checked the server will not output any text unless critical errors occur.

The IRRS settings application is similar to the IRS version; however, no database settings are needed. The application is visualized in Figure 49.

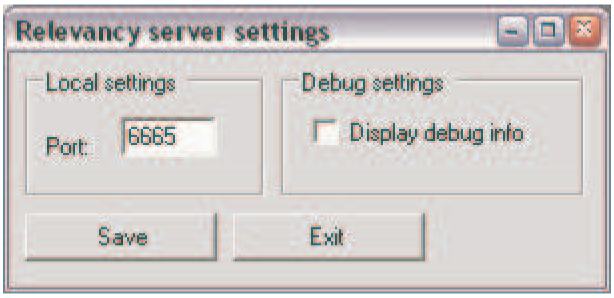

Figure 52 - The IRRS settings application

Port: The port of the incoming relevancy requests from the client. The default value for this is 6665. Any free port number could be used. NOTE: this is not supposed to be the same as the IRS port number if both servers are run on the same machine.

Display debug info: If this box is checked the server will display various debug information regarding client connections, parsed command messages etc.

Appendix E: The Configuration Files

All configuration files are in the standard .NET configuration file XML format with the <add> tag containing two attributes, key and value, the key are the name of the option and the value are the value. The internal order of these <add> elements does not matter but it is important that all different key/value pairs are present. The case of the keys and values are also important. All configuration files should also start with the <?xml version="1.0" encoding="utf-8"?> tag

IRS.exe.config

All elements are contained between the <configuration> and <appSettings> tags Ex:

```
<configuration>
  <appSettings>
    <add key=”LocalPort” value=”6665”></add>
    <add key=”DebugMessages” value=”true”></add>
    …
  </appSettings>
</configuration>
```

Key	**Value**
LocalPort	A valid port number
DebugMessages	[true|false] Print debug messages not involving SQL
MySqlHostAddress	A valid IP-address
MySqlLogin	The login name to the database
MySqlPassword	The password required to log in to the database
MySqlDatabase	The name of the database
MySqlDebug	[true|false] Print debug messages involving SQL

IRRS.exe.config

All elements are contained between the <configuration> and <appSettings> tags Ex:

```
<configuration>
  <appSettings>
    <add key=”LocalPort” value=”6665”></add>
    <add key=”DebugMessages” value=”true”></add>
    …
  </appSettings>
</configuration>
```

Key	Value
LocalPort	A valid port number
DebugMessages	[true|false] Print debug messages

XP Client – program config

All elements are contained between the <configuration> and <FormSettings> tags Ex:

```
<configuration>
  <appSettings>
    <add key=” PollInterval” value=”30”></add>
    <add key=” RelevancyServer” value=”127.0.0.1”></add>
    …
  </FormSettings>
</configuration>
```

Key	Value
PollInterval	[0-180] The time between server polls
RelevancyServer	The IP-address to the relevancy server
RelevancyPort	The listening port of the relevancy server
UrlServer	The IP-address to the main server
UrlPort	The listening port of the main server

FtpServer	The address of the FTP server which handles the annotations
FtpPort	The listening port of the FTP server
WwwServer	The full path of where the annotations will be uploaded including http:// and a trailing /. Ex: `http://www.ulve.net/annotations/`
FtpUserName	The user name used to log in to the FTP server
FtpPassword	The password used to log in to the FTP server
FtpRemoteDir	The directory in which the annotations will be uploaded including a trailing /
Uid	The user ID on the Information Radar System
Radius	The query radius in meters
SpeedDependent	[true|false] If the Radius should be speed dependent
Latitude	A decimal value representing the simulated latitude
Longitude	A decimal value representing the simulated longitude
RemoveScript	The address to the script that is responsible for removing annotations from the database including http:// and the finishing =. Ex: `http://www.ulve.net/cgi-bin/remove.py?url=`
ComPort	A valid COM port where the GPS is connected.
TempDirectory	`The path to the temp directory where all temporary files should be stored, this should include the trailing \ Ex:`

```
C:\Windows\Temp\
```

PPC Client – program.config

<configuration>
<appSettings>
<add key=" PollInterval" value="30"></add>
<add key=" RelevancyServer" value="127.0.0.1"></add>
…
</FormSettings>
</configuration>

Key	Value
PollInterval	[0-180] The time between server polls
RelevancyServer	The IP-address to the relevancy server
RelevancyPort	The listening port of the relevancy server
UrlServer	The IP-address to the main server
UrlPort	The listening port of the main server
FtpServer	The address of the FTP server which handles the annotations
FtpPort	The listening port of the FTP server
WwwServer	The full path of where the annotations will be uploaded including http:// and a trailing /. Ex: `http://www.ulve.net/annotations/`
FtpUserName	The user name used to log in to the FTP server
FtpPassword	The password used to log in to the FTP server

FtpRemoteDir	The directory in which the annotations will be uploaded including a trailing /
Uid	The user ID on the Information Radar System
Radius	The query radius in meters
SpeedDependent	[true|false] If the Radius should be speed dependent
Latitude	A decimal value representing the simulated latitude
Longitude	A decimal value representing the simulated longitude
RemoveScript	The address to the script that is responsible for removing annotations from the database including http:// and the finishing =. Ex: `http://www.ulve.net/cgi-bin/remove.py?url=`

The following configurations are exactly the same for both the XP and the PPC client

Sound.config

This configuration file is different from the other configuration files, the data is stored as elements instead of as attributes, there are an arbitrary number of <Sound> elements each with a single <Name> element which represents the file name of the sound file. All this is contained in the <Sounds> tag. Ex:

```
<Sounds>
  <Sound>
    <Name>Example.wav</Name>
  </Sound>
  <Sound>
    <Name>Example.wav</Name>
  </Sound>
```

```
  …
</Sounds>
```

Profiles.config

This configuration file is built in the same fashion as the sound.config. A <ProfileList> contains an arbitrary number of <Profile> elements and each such element should contain one of the following elements:

<Pid> - The profile ID of the profile
<Name> - The name of the profile
<Threshold> - The display threshold for the profile
<Sound> - Which sound to play (defined in sounds.config)

Ex:

```
<ProfileList>
  <Profile>
    <Pid>1</Pid>
    <Name>Philosophy</Name>
    <Threshold>56</Threshold>
    <Sound>Beep.wav</Sound>
  </Profile>
  <Profile>
    <Pid>2</Pid>
    <Name>History</Name>
    <Threshold>100</Threshold>
    <Sound>Bong.wav</Sound>
  </Profile>
  …
</ProfileList>
```