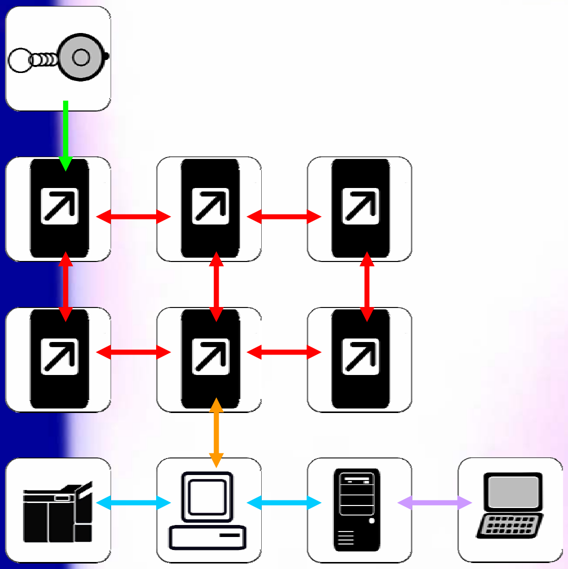
Océ

# Smart Signs for Smart Surroundings



Author: Barry Nijenhuis

Océ

# Smart Signs for Smart Surroundings

**Oce Technologies**

drs. Joost Meijer
Océ Technologies B.V.
St. Urbanusweg 43
R&D department
5900 MA Venlo

**University of Twente**

dr. ing. Paul J.M.Havinga
dr. Maria Eva M. Lijding
University of Twente
Department of Computer Science
Computer Architecture Design & Test for Embedded Systems
7522 NB Enschede, the Netherlands

**Author**

Barry Nijenhuis
University of Twente
Department of Computer Science
Computer Architecture Design & Test for Embedded Systems

Venlo 2006-01-12

# Samenvatting

Océ doet onderzoek naar onder andere een alomtegenwoordig kantoor, gebruik makend van het Océ Office Lab voor het demonstreren van nieuwe technologieën. Onderzoek dat gedaan wordt aan het office lab heeft soms ook te maken met samenwerking met het Smart Surroundings project. Deze samenwerking betreft ook mensen van de Universiteit Twente en de Universiteit van Karlsruhe. Een van de settings, die worden onderzocht in dit project, is het alomtegenwoordige kantoor. In deze setting wordt een concept genaamd "Smart Signs" uitgewerkt. De Smart Signs zijn een netwerk van draadloos verbonden apparaatjes. Deze apparaatjes helpen nomadische medewerkers door de weg te wijzen, informatie te tonen en printen te automatiseren. De Universiteit Twente zorgt voor het platform voor de realisatie van dit Smart Signs systeem. Dit platform is een netwerk van draadloos gelinkte apparaatjes, die magere bronnen (o.a. processor en geheugen) heeft en energie efficient zijn.

The hoofddoel van dit onderzoek is het vinden van de best passende realisatie van het Smart Signs systeem, gegeven het verzorgde platform. De realisatie is aangaande het semi-dynamische Smart Signs systeem, die meest van de origineel bedoelde functionaliteiten heeft, voor het genereren van gebruikers respons op de SVG Open conferentie.

Het onderzoek begint met het verkennen van de exacte intentie van het Smart Signs systeem, door het uitwerken van de eerste omschrijving tot een set van scenario omschrijvingen en een lijst van vereisten. Deze vereisten zijn verder uitgewerkt, met een meest passend Smart Signs systeem als resultaat. Dit systeem is een combinatie tusseen een gedistribueerde en centrale aanpak. Alle data hier aanwezig in een centrale opslagplaats en al het inteligente gedrag wordt geleverd door een centrale component. Ook wordt er een cache van de informatie lokaal op de smart signs opgeslagen en niet te complexe taken worden lokaal uitgevoerd. Het systeem is niet compleet gedistribueerd, omdat het gegeven platform een omgeving met schaarse bronnen betreft

Dit best passende ontwerp is ten uitvoering gebracht op de gegeven draadloze nodes. Daarna is de implementatie gebruikt voor een demonstratie op de SVG Open conferentie. De demonstratie bleek geen succes te zijn. Dit was het gevolg van technische moeilijkheden, aangaande het platform, en het te strakke schema.

Ookal werkte het systeem niet op de conferentie, is het bijna klaar en heeft het veel potentie. Daarom is het aanbevolen om dit systeem verder te testen en uit te werken, om te gebruikren voor gebruikers respons in de toekomst

# Abstract

Océ researches, amongst others, on a ubiquitous office, using the Océ Office Lab for demonstrating new technologies. Some research done at the office lab involves collaboration with the Smart Surroundings project. This collaboration also involves people form the University of Twente and the University of Karlsruhe. One of the settings, which are researched in this project, is the ubiquitous office. In this setting a concept called “Smart Signs” is deliberated. The Smart Signs are a network of wirelessly interconnected devices. These devices help nomadic workers by guiding, showing information and automated printing. The University of Twente provides a platform for the realisation of the Smart Signs system. This platform is a network of wirelessly linked devices, which are resource lean (amongst other things processor and memory) and energy efficient.

The main objective of this research is finding the best-suited realisation of the Smart Signs system, given the provided platform. The realisation regards the semi-dynamic Smart Signs system, which has most of the initially intended functionalities to generate feedback at the SVG Open conference.

The research starts with exploring the exact intention of the Smart Signs system, by working out the first description to a set of scenario descriptions and requirements. These requirements are further worked out, with a best-suited Smart Signs system as a result. This system is a combined distributed and central approach. Here all data is present at a central repository and all intelligent behaviour is provided by a central component. Also a cache of the information is stored locally on the smart signs and not too complex tasks are done locally. The system is not completely distributed, because of the resource lean environment of the provided platform.

This best-suited design is implemented on the provided wireless nodes. Thereafter the implementation is used for a demonstration at the SVG Open conference. The demonstration did not turn out to be a success. This was due to technical difficulties, regarding the platform, and a schedule that was too tight.

Although the system did not work at the conference, it is almost finished and has a lot of potential. That is why it is recommended to further test and engineer this system, to be able to use for user feedback in the future.

# Contents

# Acknowledgements

I would like to gratefully acknowledge Joost Meijer, my Océ supervisor, for his guidance, useful feedback and pleasant co-operation.

I would also like to gratefully acknowledge Paul Havinga and Maria Lijding for guiding and for their useful feedback.

Furthermore I would like to sincerely thank Tjerk Hofmeijer and  Lodewijk van Hoesel for their effort and full support, regarding the development of the smart signs.

I would like to thank all the participants in the Smart Surrounding work package 5, for their co-operation and useful observations.

I wish to thank all the colleagues residing at 3G-69 for making work at Océ a pleasant undertaking and Océ in general for making this project possible.

Finally I would like to thank my family, friends and especially my girlfriend for their full moral support and taking care of me in good and bad times.

# 1 Introduction

## 1.1 Context

Océ Technologies [OCE] offers products and services for (re)production, presentation, distribution and management of streams of documents. Océ for example supplies copy and print systems. Océ aims at a leading position in the world market, using advanced products, which distinguish themselves through high quality, reliability, productivity, durability and user- and environment friendliness. The assortment is mainly developed and produced by Océ itself.

Research and Development is the branch of Océ that works on the development of new technologies and new products derived from these technologies. This branch researches and works for the future of Océ. One topic that Océ R&D researches is that of a ubiquitous office. This vision is materialised in the "Office Lab"[HSSN], in which Océ researchers can show their prototypes, which fit this vision.

Some research done in the Office Lab also involves collaboration with other parties. One example is the collaboration in the Smart Surrounding project. This project has the mission to investigate, define, develop, and demonstrate the core architectures and frameworks for future ambient systems. The Océ Office Lab is used, in this collaboration, as a setting to make concrete prototypes and to retrieve user feedback. In this settings subgroup (Work package 5 of Smart Surroundings) people from the University of Twente and the University of Karlsruhe are also involved. [SSUR]

In the vision of this office setting, the idea of a nomadic worker takes a central place. The physical location of a nomadic worker is irrelevant for access to work content and contact with other people at anytime. To substantiate this concept, two settings were worked out. One of these settings is the "Flexible office", where every office worker has means to work from any place within a company as well as from another location. For this setting several scenarios were created. [WP5S]

One of these scenarios describes the nomadic worker being in a flexible office, using a system of "Smart Signs". The Smart Signs, on the office doors, help the nomadic worker to work in the flexible office by giving room information, personalised information and by guiding the person through the nomadic office. The Smart Signs scenario contains the basic thought behind this report.

In this report the realisation of the Smart Signs on a platform provided by the University of Twente is discussed. This platform consists of two parts: hardware and software. The hardware part is the "sensor node" [NSN], which the system should use as the signs. The software part of the provided platform is the operating system AmbientRT [ART] [AMAN], on which the Smart Signs software has to run.

## 1.2 Objectives

The global objective of this report is to get an answer on the research question, which is:

**What is the best-suited realisation of the Smart Signs system, given the provided platform?**

Apart from answering this question, there are more objectives. Now the involved parties with their objectives are listed.

### 1.2.1 Personal objectives

- Gain more knowledge about embedded devices.
- Gain experience working with embedded devices.
- Learn more about designing and realising a product.
- Apply gained knowledge.
- Gain experience on working in a company. (~ time pressure, colleagues, real world ~)

### 1.2.2 Océ technologies

- Gain knowledge about new technologies and office concepts.
- Display a demo at the Océ Office Lab.

### 1.2.3 Smart Surroundings

- Gain knowledge about the realisation of the Smart Signs system.
- Show a demo of the Smart Signs at the SVG Open conference [SVG].

## 1.3 Approach

My approach for answering the research question is to first gain more information about the characteristics of the hardware and software platform, on which the Smart Signs system should be operating. Thereafter the high level design is made, deciding what solution, to various design problems, is best suited in this context. When the high level design is finished, the system is implemented. Finally the system is put to the test, by testing and a demonstration.

## 1.4 Report Structure

This report will discuss the design process of the Smart Signs system. First the methodology will be discussed. Here the used instrumentation and methodology are discussed. Also the planning of the project is discussed in this chapter. Thereafter the design and implementation are designed, describing the decisions made, the problems encountered and the descriptions of the various system components. After that the tests and the demonstration will be discussed, describing the validation of the designed system. Next the results of the implementation discussed, describing the interpretation of the results. Finally the conclusions and recommendations are given, to answer the research question and to discuss further work and other conclusions taken, regarding this report.

# 2 Instrumentation and Methodology

Before exploring the exact meaning of Smart Signs, first the instrumentation and methodology is described. This chapter first describes the used tools (while implementing). Thereafter the used diagram categories, used for modelling the system, are discussed. Finally the model legend, which gives the explanation of the used pictograms, is given.

## 2.1 Used tools

The used software tools for implementing the Smart Signs system is different for every part of the subsystem. The only “tool” that is used in every subsystem is the programming language C (or a variety on C). Hereunder the subsystems are listed with their used tools.

- *Central server:* “Vim”, for editing the code and “gcc” for compiling the code.
- *Gateway:* “Crimson editor” [CRIM] to edit the code and “gcc” in combination with “MinGW” and “MSYS” to compile it.
- *Smart Signs:* “Crimson editor”, to edit the code and “mspgcc” in combination with “MinGW” and “MSYS” to compile it and flash the program into the memory of a smart sign.

## 2.2 Used diagrams

While designing the Smart Signs system, a set of diagrams was used to specify the system. These diagrams are used, because they have several advantages: They give overview, the diagrams make it easier to explain and understand a system, and they let one think about the system on a high level. Every type of diagram used for describing the system has their own characteristics and therefore their own advantages. Hereunder the used diagrams are described with their characteristics and advantages.

### 2.2.1 Data Flow Diagram

Data flow diagrams (DFDs) are used to give a clear picture about the (static) architecture of the system. These diagrams show whether data is exchanged between different processes. This gives a good view of what components are present in the system (high and low level components) and how they are related to another. The DFDs are available in Figure 9 and 10. (for more detailed information about DFDs, please read [AMOD]).

### 2.2.2 Sequence Diagram

While DFDs show the processes and their relations to each other, the Sequence Diagrams (SD) gives more information about the stream of data between the processes. A SD depicts the sequence of actions that occur in a system. This gives a good overview of the dynamic behaviour of the system and shows how the data is passed from one process to another. The Sequence Diagrams of the system are available at appendix G. For more information about Sequence Diagrams, please read [AMOD].

To get a more descriptive and graphical picture about the flow of data a variant on the SD is used. These graphical diagrams are used in chapter 7.

### 2.2.3 Use Case Diagram

In addition to showing the internal behaviour using DFDs and SDs, also Use Case Diagrams (UCD) are used to show the external behaviour of a system. With this diagram one can see all the available functionality of the system at a glance and a very high level. This gives a good black box view of the system, where one can see all the system functionality, while not having to dig though details. The Use Case Diagrams are available at chapter 3. These UCDs have a little bit more information added to a normal UCD. Here one can also see the environment objects, which the system is connected to. For more information about Use Case Diagrams, please read [AMOD].

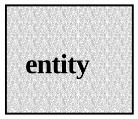
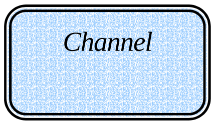
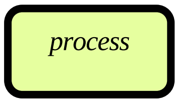
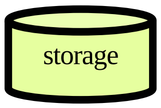
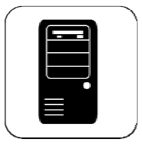
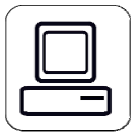
## 2.3 Used planning

Appendix A and B show the planning used for this project. Here only the differences between the initial global planning and the interim planning, and the differences between the initial planning and the actual execution will be discussed.

The reason behind the creation of the interim planning is that, when enough information was collected at the university, it became clear that the development of the static system would not be sufficient to get proper evaluations. Also the gap between the static and the dynamic system (see chapter 3) would be too big. That is why the idea of the semi-dynamic (see chapter 3) system came to mind. This made a change of planning necessary. For further explanation please see chapter 3.

There are also differences between the interim planning and the actual execution (marked bold). These differences have two reasons. The first reason is that some deadlines included a safety buffer to get everything done for sure. The second reason behind the differences, is that of an unforeseen development at the (preparation of) the conference. This is discussed in detail in chapter 9.

## 2.4 Model Legend

| Component | Explanation |
|---|---|
|  | This grey filled square represents a high level entity. This entity represents a main component of the Smart Signs system. |
|  | This rounded rectangle represents a communication channel between the high level entities. Through this channel the high-level entities send data to each other. |
|  | This rounded, green, rectangle represents a process in a Data Flow Diagram. This process can communicate with other processes and storage facilities. |
|  | This symbol represents a repository, where information can be stored. It is used in Data Flow Diagrams and can communicate with processes. |
|  | This sign represents a central server, which can be used in the Smart Signs system. |
|  | This sign represents a gateway computer, used in the Smart Signs system. |

| | |
|---|---|
| 💻 | This sign represents a client (for example a computer or handheld) which a user can use to communicate with the Smart Signs system. |
| 🖨 | This sign represents a printer, where the Smart Signs system can communicate with. |
| ↗ | This sign represents a smart sign in the Smart Signs system. |
| ◎ | This sign represents a tag, which is an element of the Smart Signs system. |
| ▣ | This high level sign represents the whole Smart Signs system, which can be used to describe the system at a high level. |
| 🧍 | This symbol is used to represent a user of the Smart Signs system. |
| → | This red arrow represents communication between two smart signs. |
| → | This blue arrow represents communication between two computer components (also printer). |
| → | This orange arrow represents communication between a smart sign and a computer. |
| → | This purple arrow represents communication between a guest computer and a central server. |
| → | This green arrow represents communication between a tag and a smart sign. |

# 3 Scenarios

This chapter discusses the elaboration of the scenarios. This process starts with the first and second draft. Thereafter the first set of scenario descriptions (static and dynamic) is given. Then a revision is described. Finally the set of scenarios, used for further design, is elaborated.

## 3.1 First draft

The very first draft is partly derived from a report of the Smart Surroundings group (Work package 5) [WP5S]:

*"The Nomads also offers a system of '**Smart Signs**', which the employees can use at the doors of the office they occupy. The Smart Sign shows the name of a person working in a particular room and some information of where the employee currently is or when he is expected to be back. It can also show some personalised message addressed to a specific passer-by"*

…………………

*"As she changes her offices so frequently, Martha never knows where the closest office facilities such as printers are. Mini-Martha helps her to deal with it with a new use of the '**Smart Signs**'. The Smart Signs can collectively show to Martha the way where she should pick up her prints. These are arrows projected in the direction of the printer, or she can get the route on her personal handheld device."*

From this quoted text two functionalities of the Smart Signs can be extracted:

- Showing general and personalised room information.
- Guiding persons to locations in an office.

The following functionalities are added, as a result of a meeting in the Smart Surroundings workgroup:

- Make printing easier.
- Exchanging personal information between two people.

These four functionalities contain the core of the first draft.

## 3.2 Second draft

The second draft is the result of a discussion about the feasibility of designing the system all at once. This discussion concluded that the project should be split into two stages. The three main reasons for splitting the project into two phases are:

- The system to be designed needs to have an operable prototype for the SVG Open 2005, a conference and exhibition from the 15th until the 18th of August 2005. To have the whole system running before this date is an impossible task.
- Splitting the project into two phases means splitting into more bite-size pieces to develop.
- At the conference the system can be put to the test and the obtained feedback can be used for developing the final system.

The first phase is the static approach of the system, where the smart signs does not have any communication at all with its environment. This system will be regarded with the name "Static system" throughout the report. This static system has the following functionalities:

- Showing general room information
- Guiding persons, without knowing where they are (showing the guide information on all the smart signs).
- Make printing easier, by guiding to a printer after a print request is made.

The second phase is that of the full system ("Dynamic system"), as described in the first draft. This includes communication with, devices which represent, persons.

## 3.3 Scenario descriptions

The first draft is very general and can result in many different systems. That is why scenario descriptions, in combination with Use Case Diagrams, are used. They give a more concrete description about the purpose the system will be used for. They also mark out the to be developed system. The scenario descriptions are a set of small stories, which informally describe the system to be developed.  First the static and dynamic scenarios are cited. Thereafter the revision, which leads to the semi-dynamic system, and the final system are discussed. This final system will be described using scenario descriptions and Use Case Diagrams.

### 3.3.1 Static system scenario descriptions

#### Guiding Chris

Chris comes to visit his own flexible office and checks in at the reception. Chris has never been at that building before and therefore does not know the way to his room. He asks the way to his own office at the reception. The receptionist types the request into the computer system and tells Chris that he has to follow the signs. Chris sees a small flat sign in the direction the receptionist has pointed. On the sign an arrow points to a direction. After following multiple signs, he ends up at the room where he has to be.

After working for a while he suddenly remembers that he has to ask his colleague Bill something. Because they are in a flexible office, Chris does not know where Bill is working today. Thus he grabs his PDA and requests guidance to Bill's room. Again he sees that a path is shown on the signs on the walls. He follows the signs to Bill's room and asks him the question.

#### Printing for Pete

Pete is at a conference and sits behind a computer. He sees an interesting paper and wants to print the document. After pushing the print button, a message appears on the screen indicating that the signs on the wall will indicate his path to the nearest printer. Pete spots a small flat sign close to him. It shows an arrow and information about his print. He looks further ahead and sees other signs, all pointing in a single direction. He decides to follow the directions and already hears a printer printing documents close from where he is. When he reaches the printer, he is pleasantly surprised, his paper has already been printed.

#### Knocking on the door

Bam is at a conference and wants to attend a lecture, but is a little bit late. On top of that he cannot find his program. He thinks the lecture is in room1. While walking to room1 he comes past room2. A small flat sign on room2 indicates information about what is going on in room2. While quickly reading the information on that sign he sees, to his surprise, that the lecture he wants to attend is at room2 and not at room1. Bam is relieved that he finally found the lecture he wanted to attend and enters the room where the lecture is taking place.

### 3.3.2 Dynamic system scenario descriptions

#### Guiding Chris

Chris comes to visit his own flexible office and checks in at the reception. Chris has never been at that building before and therefore does not know the way to his room. He asks the way to his own office at the reception. The receptionist gives a small sign to Chris and tells him that the sign he just received will guide him. Chris walks further into the building and notices a sing on the wall, which shows an arrow. He follows the direction of the arrow. When he comes close to another sign on a wall, again an arrow appears. After following all the directions, he ends up at the room where his flexible office is.

#### Printing for Pam

Pam arrives at a conference. While checking in, to receive program information and such, she receives a small sign. The guy at the desk explains him that she can use this device to be guided in finding his way and for printing. Pam does not know the building where the conference is being held, so she takes the sign. She wonders what the sign for him could do regarding printing. After attending some lectures, she decides to look on the Internet to get interesting information about a lecture that she just had at one of the public computers. When finding an interesting paper, she decides to print it. She soon sees on his sign that she has a print job pending. Pam also sees directional information on the signs on the walls. She decides to follow the directions and finally arrives at the printer where his print job(s) are already printed.

**Knocking on the door**

Bam is at a conference. He has nothing to do and does not know the exact program of the following hours. That's why he decides to just walk up to a room and see what is happening there. When he arrives at room1 he sees a small flat sign on the door, which indicates some global information and additionally shows more detailed information for Bam, that his colleague is going to have a speech there. Then another person walks up to the door and the personal information on the small flat sign disappears. When he looks to his own sign, he sees the personal information he just saw on the sign on the door. Finally Bam decides to enter the room and attend the lecture.

**Socialising**

At a conference Bam and Bob both got a sign at the beginning of the conference. They had to give their information (personal details, interests, etc) to the person, which gave them the small flat sign. While Bam walks to the coffee machine, where Bob also stands, his sign gives information about Bob (contact points of interests for example). While viewing this, he sees some interesting information and decides to begin a conversation with Bob, while both having some coffee.

### 3.3.3 Scenario revision

The scenarios of the static and dynamic system were discussed on the 20th of April 2005 at a Smart Surroundings (Work Package 5) meeting. The results of that meeting again changed the design of the Smart Signs system. It was concluded that the dynamic system did not need any change. However, the static system needed change. The static system was first intended to have an intermediate system and to get user feedback. The usage of the static system for generating feedback turned out to be insufficient, it has too less functionality. The biggest shortcomings of the static system, regarding fetching user feedback on guiding a person:

- All the signs display guide information for a person, this makes guiding multiple persons at a time difficult. It gets too confusing and too much of an effort to extract guide information from a sign, especially from such small signs.
- Only a very limited amount of users can be guided, when overview needs to be kept.
- The signs, which are going to be used as the hardware platform (see chapter 5), will be completely static in this system. These nodes are not designed for this static kind of application.
- There is no kind of localisation whatsoever.

Because of the lack of functionality of the static system, regarding the purpose to get feedback, the decision was made to develop and use a "semi-dynamic system", instead of the completely static system. This semi-dynamic system has the same characteristics as the static system. The only difference between the static and the semi-dynamic system is that the semi-dynamic system includes a so-called "tag", which represents a user. Now the system can approximate the location of a user. This results into signs only showing guide information, when a user is in the proximity of that sign. Now more users can be guided at the same time, without loosing overview. An extra advantage of using a "tag" is that the system is now able to show personal room information and enables personalised printing (as described in the dynamic system). The semi-dynamic system is discussed in more detail below, using scenario descriptions and Use Case Diagrams.

## 3.3.4 Semi Dynamic system scenario descriptions

### Knocking on the door

John is at a conference. He has nothing to do and does not know the exact program for the following hours. That's why he decides to just walk up to a room and see what is going on there. When he arrives at room1 he sees a small flat sign on the door, which indicates some global information and additionally shows more detailed information for John, he sees that his colleague is holding a speech there. This makes him decide to enter the room and attend the lecture.

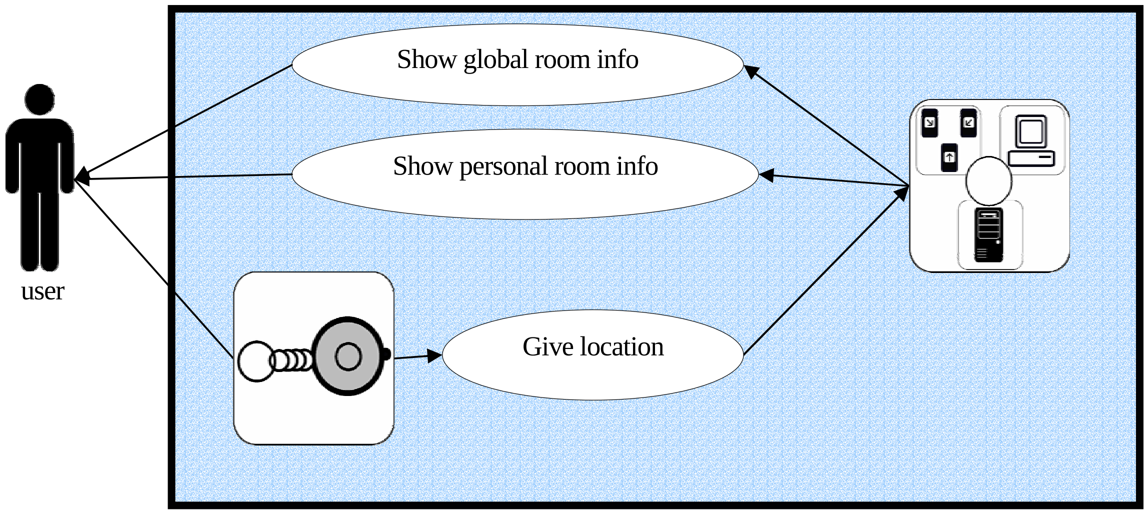


*Figure 1 : Use Case Diagram; showing information*

### Guiding Chris

Chris comes to visit his own flexible office and checks in at the reception. Chris has never been at that building before and therefore does not know the way to his room. He asks the way to his own office at the reception. The receptionist gives a small token to Chris and tells him that the token will help guide him. Chris walks further into the building and notices a sign on the wall, which shows an arrow. He follows the arrow. When he comes close to another sign on a wall, again an arrow appears. After following all the directions, he ends up at the room where his flexible office is.

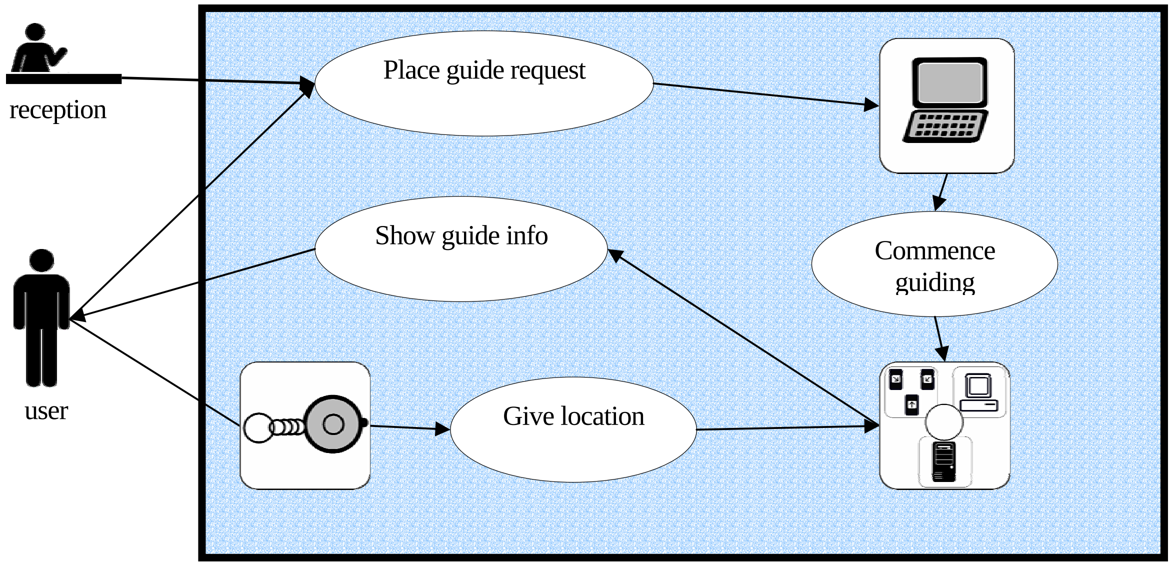


*Figure 2 : Use Case Diagram; Guiding*

## Printing for Pam

Pam arrives at a conference. While checking in, to receive program information and such, she receives a key chain. The guy at the desk explains him that she can use this device to be guided in finding his way and for printing. Pam does not know the building where the conference is being held, so she takes the sign. She wonders what the sign for him could do regarding printing. After attending some lectures, she decides to look on the Internet to get interesting information about a lecture that she just had at one of the public computers. When finding an interesting paper, she decides to print it. She is informed that the signs on the wall will help him find the printer. Pam sees the directional information on the signs on the walls. She decides to follow the directions and finally arrives at the printer where his print job has already been printed.

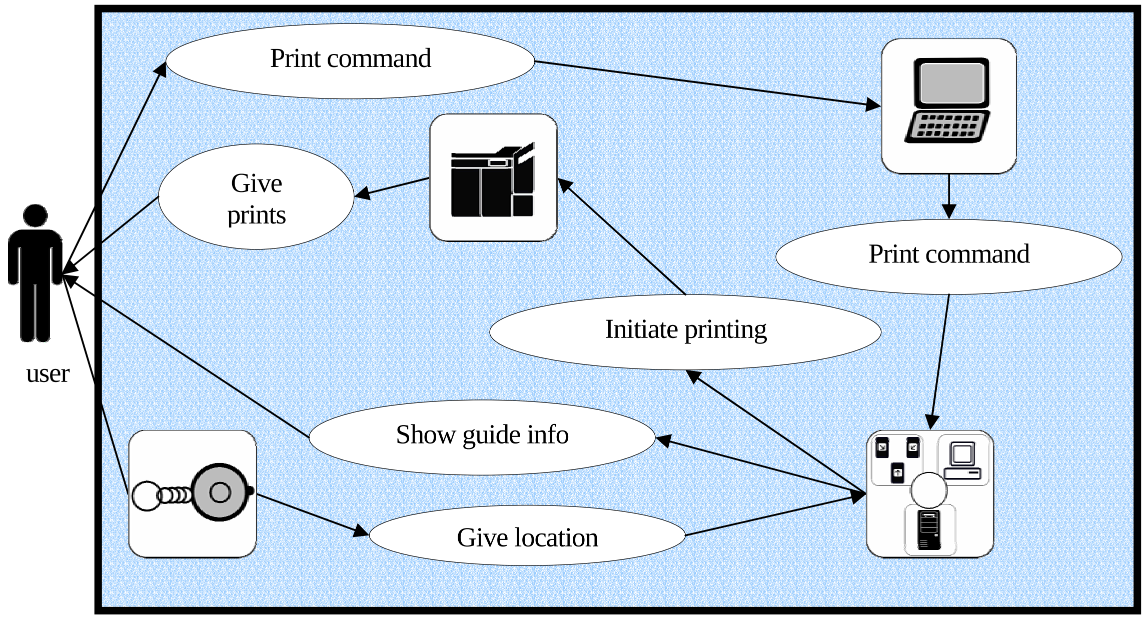


*Figure 3 : Use Case Diagram; Printing*

# 4 Requirements

The next step of the design process is to further elaborate that what is informally written down at the scenario descriptions and what has been discussed during meetings. The requirements do just that, they are a list of things the system should do. The requirements are grouped into sections, to relate the requirements to the different scenario descriptions. The requirements of the static and dynamic system can both be found in appendix C and appendix D. These requirements are from before the scenario revision mentioned before, to give a good perspective of where the semi-dynamic system stands. The requirements of the semi-dynamic system are given below. These requirements serve as a basis for the rest of the design and implementation process.

## 4.1 Guiding

1. The network of signs has to provide the facility to place requests for guidance at a laptop/handheld device.
   - 1.1. The network of signs has to provide a way to communicate with external devices.
   - 1.2. The network of signs has to be able to compute a requested destination to directions on nodes.
2. There needs to be a way to distinguish a sign, which represents a room.
3. There needs to be a way to distinguish a tag, which represents a person.
4. The network of signs has to be able to show personalised directional information.
   - 4.1. The network of signs has to be able to approximately determine the position of a tag, which a user carries
   - 4.2. The network of signs has to be able to show personal directional information on signs in the proximity of a person.

## 4.2 Printing

1. There need to be a way to distinguish a tag, which represents a person.
2. There need to be a way to distinguish a sign, which represents a printer.
3. After giving the command to print, the network of signs has to show personal print information.
   - 3.1. The network of signs has to be able to approximately determine the position of a tag, which a user carries
   - 3.2. The network of signs has to be able to show personal print information on signs in the proximity of a person.
   - 3.3. The network of signs has to be able to manage prints of users, to make printing location specific.
   - 3.4. The network of signs has to be able to track the requests for prints users make.
4. The network of signs has to be able to activate print jobs to a specified printer, according to a location.

## 4.3 Showing information

1. Signs, which represent rooms (situated on for example the door of that room), have to be able to show general information about that room.
   - 1.1. All signs need to be able to display information.
   - 1.2. There need to be a way to distinguish a sign, which represents a room.
   - 1.3. The network of signs has to be able to distribute location specific information to specific signs.
2. The signs in the proximity of a person need to show personal room information, when that person is close to the mentioned room.

# 5 Technical overview

Before one can take decisions about the architecture of the system, one first has to know the platform on which the Smart Signs architecture needs to be situated.

## 5.1 Hardware platform

The hardware, which the Smart Signs system uses, consists out of two main components. The first component is a personal computer. This is a commonly used piece of hardware on where the calculation intensive jobs are done. The important characteristics for the used personal computers are the following: (see chapter 7 for the explanation of the components)

- *Central server:* 400 MHz processor, with 128 MB memory.
- *Gateway:* 3GHz processor, with 512 MB memory.

### 5.1.1 Smart Signs

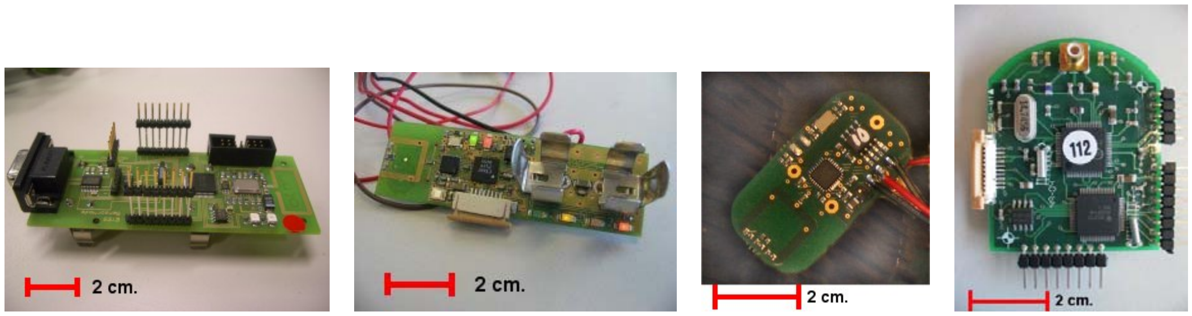


Figure 4 (fltr): initial smart sign, latest smart sign, tag and display node

Apart from the personal computers, a network of multiple “smart signs” also provides part of the hardware platform. This part of the hardware platform needs to be able to:

- Last long
- Connect wirelessly
- Show information
- Process data.

The provided network of smart signs satisfies the above mentioned points, because they are designed to be as energy efficient as possible. The signs are also wirelessly connected, can show information and process data. The only thing that needs to be taken into account is that the smart signs are resource lean (to spare energy). This takes its toll on the performance of the smart signs. Memory, energy and processing power are scarce in this environment. While designing and implementing this has to be taken into account. The smart signs are also chosen, because there was a need for such a platform and the University of Twente, which participates in the Smart Surroundings project, has this technology.

To give an idea about the characteristics of the different components; the processors runs at 8 MHz and have 2KB RAM and 60KB+256B flash memory (see [MSP4] for the initial smart sign and the display node and see [MSP6] for the latest smart sign and the tag). Every component, except the tag, also has an extra (eeprom) memory of 256 KB. The radio of the initial smart sign runs at 868 MHz [RFM] and the radio of the latest smart sign and the tag operate at 433,868 and 915 MHz [NOR]. The speed of the radios can go up to 115.2 KBPS. The initial smart sign and the display node also have the ability to connect to a LCD display [EMIC].

Through these, above mentioned, characteristics and the comparison with the personal computer, one can see that the smart signs are very resource lean. The tag has even less resources to its disposal.

### 5.1.2 Communication

The Table below shows the communication speeds, between different hardware components. Please note that the communication speed from one node to another is ver slow, because the speed is already slower than the rest of the communication and a smart sign only has one slot per second. Please note that the sensing of a tag does not affect the communication between two smart signs and is not involved in this type of communication.

| Communication type | speed |
|---|---|
| Between computer  and smart sign | 19200 baud (19200 bits per second) |
| Between two computers | 500 kilo bits per second |
| Between two smart signs | 32 (slots per second ) * 128 (data size) = 4096 byte/s |
| Number of messages per smart sign per second | 1 message per second (1 slot per sign, each second) |

## 5.2 Software platform

The Smart Sign system is build on top of existing software. This software platform consists of an Operating System  (with some additional software, regarding the gateway), which is different for every component, namely:

- *Central server:* Linux
- *Gateway:* Windows 2000, with additional software MinGW, MSYS [MGW] and mspgcc [MGC].
- *Smart Signs:* AmbientRT [ART].

The first two operating systems are commonly known. The last one (AmbientRT) is not known so well. This operating system is developed by the University of Twente and is custom made for the smart signs hardware platform.
The additional software used with Windows 2000 is needed for development of the Smart Sings system. MinGW and MSYS are used to create an environment where one can use GCC (Gnu C Compiler) to compile programs. Mspgcc is a compiler for the Texas Instruments MSP430 family (processor of a smart sign). Using this software one can compile a program tailored for a smart sign (see hardware platform) and load (flash) it into the memory of a smart sign.

# 6 Design decisions

The requirements state what the system has to be able to do. This does not tell anything about what the design should look like. Here the design choices, which will lead to the final architecture, are discussed. The basis of this process of choices are the requirements and the scenario descriptions. Therefore, the same structure as the scenario descriptions will be used. First the overall design choices are discussed and thereafter the design choices per scenario (group of requirements) are debated. All the design choices explain the following things:

- Explanation of the choice to be made
- Listing Advantages and disadvantages of each available choice.
- Weighing the characteristics and describing the selected choice.

Before discussing the decision process, first the hardware components, which are required, are named. This quickly summarises the context in which the decisions have to be made. Also an example situation is given, which is used to quantify the proportions of different options.

### 6.1.1 Hardware components

Regarding the requirements several hardware components are certainly needed in this system:

- Smart signs, which are certainly needed to display information to users.
- Personal computer(s), which are surely needed for handling guide requests and handle print requests.

These two components can also be used for other purposes, but can potentially be fulfilled by both hardware components. For more information about the hardware platform, please see chapter 5.

### 6.1.2 Example situation

Here a realistic example situation is described in a short table. This example is realistic, because all the characteristics are real life characteristics, which can be applied to any real smart signs network. The important facts are stated, which allows outlining the quantities of the different options, described later on. The calculations are done concerning the worst case scenario, where a sign is 4 hops away from the nearest gateway sign. A maximum of four hops is realistic, because this is also the maximum hops in the network used for the SVG conference (please see Figure 18 for more information). Figure 6 graphically describes the worst case scenario, where the rightmost sign is used for calculations.

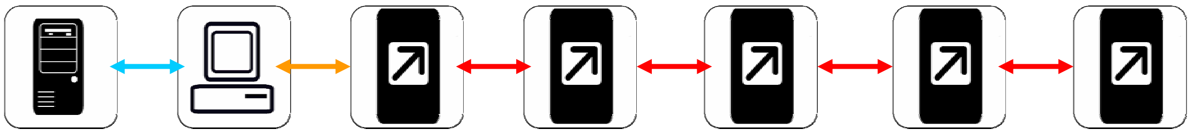


*Figure 5 : Example situation*

| Overall characteristic | Quantity |
|---|---|
| Number of smart signs | 50 |
| Number of users with personal information | 50 people |
| Maximum number of hops of a sign away from a gateway | 4 hops |
| Maximum message size | 32 bytes |
| Guide message size (communication) | 10 bytes → 6 bytes header (type, tag id, route) (static); 4 bytes info (1 byte direction, 1 separation byte (static), 2 bytes extra info (“up” for example, can become more) (dynamic). (<= 1 smart sign communication message) |
| (node, print) Information message size (communication) | 20 bytes → 6 bytes header (static); 14 bytes info (for example: “this is a test”) (dynamic) (<= 1 smart sign communication message) |

| **Storage characteristics** | **Quantity** |
|---|---|
| Node and Print Information size | 7 bytes → 3 bytes header (user id, age, info length) (static); 4 bytes reference to info on flash memory (static) (with usage of flash memory) |
| Guide information size (minimal) | 3 bytes → 3 bytes header (user id, age, direction / destination) (static) |
| Smart sign main memory | 200 bytes |
| **Timing characteristics** | **Quantity** |
| Timing smart sign←→computer communication (guide) | (10 bytes * 8) / 19200 = 4 milliseconds |
| Timing computer communication (guide) | (10 bytes *8) / 500000 = 0.16 milliseconds |
| Timing smart sign communication (guide) | 1 second (1 message per second) |
| Timing smart sign←→computer communication (info) | (20 bytes *8) / 19200 = 8 milliseconds |
| Timing computer communication (info) | (20 bytes *8) / 500000 = 0.32 milliseconds |
| Timing smart sign communication (info) | 1 second (1 message per second) |

*Table 1: Relevant information example situation*

A small note concerning Table 1 is that one can see that, although the smart signs network speed is 4096 byte/s, only 1 message per second can be send from one sign to another. This is because the communication is divided in slots, where different signs can send something. A sign only has one slot per round (second) and that, for example, means that in a network of four hops, a message takes 4 seconds to come across. Another note is that personal information is personalised information about the environment of a smart sign that only applies to a single user. For example a sign can display a message “busy until 3”, for one user and the message “available now” for another user. Finally the storage of guide information can be done the minimal way

Other facts, like the time to process information (on a smart sign and computer) are ignored in this example, to keep the calculations and the overall picture simple. A sign with the maximum amount of hops away from any gateway is chosen as a reference smart sign, to use in the calculations.

### 6.1.3 Displaying Information

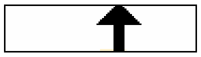
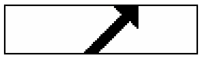
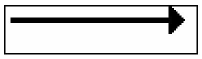
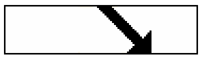
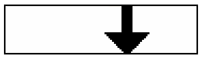
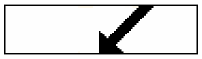
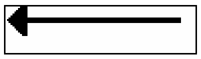
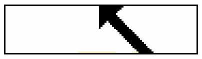
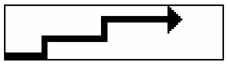
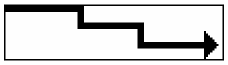
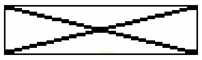
A design choice that concerns all the three scenarios, is how to actually display the information on the sign. The information can be (personalised) object (for example a room) information, guide information and print information. The available platform on which the data needs to be shown is a small LCD display. The (personalised) object information and the print information are mainly textual information. Guide information can be represented in various forms. The challenge is to put as much information on the screen, without loosing overview.

The characteristics of the LCD display are a big factor to take into account. The screen is very small, has a low resolution and is also not lit. This results in the fact that the overview easily gets lost. The balance has to be found between the number of users being served with information and the overview. Too many users with information will take its toll on the overview. For example, it can become very difficult to distinguish which data is for whom. The best overview, on the other hand, results into showing only general information or information for one user. For example, showing one big arrow for directions. This is not the behaviour that is sought after. This will be too limited for areas with multiple users.

The best balance is found by segmenting the screen into a number of segments, which keeps the overview for as many users as possible. In these segments one type of information for one person can be shown. These segments are dynamically allocated. For example, when one person is near a sign, then multiple segments will be for that user, which can show guide, object and print information (when available). When there are multiple users near the sign, then a segment is for one person, which shows the most important information. The most important information to be shown is the guidance information. This is because when a user does not get the directional information, then the person does not know where to go and gets lost. The other information is also good information to know, but not critical. It is chosen to do the guiding by arrows. The reason behind this is that a picture is more intuitive than a textual description. It also gives more overview and is easier to see from a distance.

There can also be different types of arrows for different users; to make it even easier to spot the arrow meant for one particular user.

For this display, the number of segments was chosen to be three. This means that a maximum of three users can be served with either object, print or guide information. Hereunder the chosen set of signs is shown. This is the set used at the SVG conference (see chapter 9).

| Sign | Abbreviation | direction |
|---|---|---|
|  | U | Up |
|  | UR | Up right |
|  | L | Right |
|  | DR | Down right |
|  | D | Down |
|  | DL | Down left |
|  | L | Left |
|  | UL | Up left |
|  | SU | Stairs up |
|  | SD | Stairs down |
|  | E | Error : this path should not be avaialble |
| | OL | Océ logo : initial screen |
| | UL | UT logo : initial screen |

### 6.1.4 Central versus Distributed

The biggest design choice to be taken is whether the whole system has to be designed in a central or a distributed way.
This choice is about what the responsibilities of every component should be, which involves the question where the data should be stored and where the processing should be done. Both the central and distributed approach has its advantages and disadvantages. These are deliberated below.

**Central approach**

| *advantages* | *disadvantages* |
|---|---|
| Data is easy to manage (data in central place). | A lot of overhead communication between central server and the smart signs and smart signs. |
| Complex tasks are easily done on heavy-duty central server. | Simple tasks are done relatively slow, because of the communication overhead. |
| The signs can stay simple. | Everything relies on the central server |
|  | Slow response time. |
|  | Privacy, users are tracked. |

*Table 2 : Advantages and disadvantages of a central environment*

Applied on the Smart Signs system, it means that all the information is stored on a personal computer. This generates a lot of communication messages, which are very costly in energy terms. In this context the energy is very scarce, so without having some kind of cache, this would mean that the energy drains very quickly. All these messages are also very time consuming; this definitely affects the response time of the system. With the central approach the response time will be too slow, because every time the desired information needs to be fetched from a central personal computer. Also when a sign in the network fails, then it could be possible that whole parts of the system can not be reached anymore.

**Distributed approach**

| *advantages* | *disadvantages* |
|---|---|
| No dependence on a single link in a chain. | The smart sign has to perform complex tasks |
| Only tag to node communication | Data management, over multiple instances, is complex |
| Privacy issue can be avoided. | The smart sign need large storage capacity and relatively complex processor. |
| Fast response time. | |

*Table 3 :Advantages and disadvantages of a distributed environment*

Applied to the Smart Signs system, a distributed solution means storing all the data regarding a node on the node itself. This involves storing routing information, to all other smart signs, and all object data (general and personal) on the sign itself. Storing extra information on the tag is also required. In this situation only communication between a smart sign and a tag is necessary.
A positive side is that the smart signs do not waste that much energy, because communication soaks up most of the energy. However, this approach would require a lot of storage capacity from the node. The smart signs, on which the product will be running, have limited memory and process power. There is only 2 Kilobits of internal memory and 2 Megabits of flash memory. The flash memory is not the biggest bottleneck (although it is very slow), but the internal memory is very small and has to have an OS (AmbientRT) in it. Also all data, which is stored, has at least a pointer in the internal memory, which will take at least 1 byte of the memory. This all makes storing all the node specific information on the node itself impossible. Node specific information can be room information, but also personalised room information, guide information for every user in the system (which is intended to be a lot) and a physical structure of (a part of) the system (for guiding purposes) needs to be stored. Regarding the processing of information a distributed solution is also not desirable. Processing everything locally means doing complex tasks on the smart signs itself. The smart signs are a resource lean environment and not designed to perform (a lot of) complex processing.
Another upside of this distributed approach is that there is no privacy issue. However, the tag is in this context too recourse lean to store data on. Here it only acts as a beacon. Therefore a fully distributed option is not feasible.

## Chosen

The selected choice is to do the data management and processing in a combined central and distributed manner. In this way most advantages of both sides are used. This solution involves storing general information and as much personalised information as possible in the memory lean signs. To begin with, all information is stored in a central server. The signs have, in addition to the general information, a local cache of personalised information, which is filled with the most recent information. Whenever a smart sign senses a new person and the sign does not have information for that person, relevant data from the central server is send to and stored on the sign. Also a variety of flooding is used to send as much, of dynamic (personalised) information for storage, in advance to the sign as possible. This will, for example, improve the response time. But still the heart of the system remains the central server, which takes care of the entire complex process and memory intensive work. This combined solution does not need a complex algorithm for data management and the signs have, to some extent, as much information as possible stored locally.

A disadvantage of this approach is that the system depends on the central server, but still less than the completely central approach, where no information at all is stored in the signs. This disadvantage does not weigh up to the disadvantage of having to come up with a very complex data management algorithm and to have a lot of communication overhead, which is impossible in a resource lean environment such as the smart signs. Another disadvantage is the still present privacy issue. However, if circumvented wisely, one can neutralise this disadvantage. This can be done in many ways; one can for example let the user decide, until what privacy level the information is stored. One can also put a button on the tag, whereon the user can switch the tracking off.

### 6.1.5 Guiding

While being guided, some guide information should be shown to the user. The question is how should this be stored and managed? Which solution is best suited for the given environment? Hereunder the three available options are weighted.

#### Full store

The first option is the "full store" approach, where all guide data is available on every sign. Physical location information is stored in each and every sign (minimal1 byte per direction for every location), so that in every case the sign knows what the corresponding direction is. It furthermore stores the guide information (3 byte per guide, containing the user id and destination) in the memory. Then when a person comes by, who want to go to a certain place, the sign chooses the right direction to show. The information is given to every sign (broadcast), when a guidance is placed. This option also requires an initial setup of the network, namely sending the appropriate routing table to every smart sign. This involves sending two messages per smart sign (50 bytes divided by the maximum message size).

Applied to the example situation, the following characteristics are calculated:

Initial setup $= 50\ signs \times 2\,messages = 100\,messages$
Storage space (only routing information) $= 50\ signs \times 1byte = 50\,bytes$
Storage space (all users guided) $= 50\ signs \times 1byte + 50\,users * (3bytes) = 200\,bytes$
Smart Sign communication messages $= 1 \times 4\,hops = 4\,messages$
Reaction time (after broadcast) $= 0\,\text{sec}.$ (no communication)

#### No store

The second option is the "no store" approach. Here the information is send from a server to a sign on demand. First a request for guidance is made at the central server, through a connected computer. Then a path is generated and being stored, to be used later. When a person passes by, the sign asks the server for information. The central server thereafter sends appropriate direction information to the sign in question, which shows the direction to the person. When a person is not sensed anymore, the sign discards the information.

Applied to the example situation, the following characteristics are calculated:

Storage space $= 3\,display\,components \times 7\,bytes = 21\,bytes$
Smart Sign communication messages $= 2 \times 4\,hops = 8\,messages$
Reaction time $= 8\,messages \times 1\text{sec}. + 2 \times (4\,ms. + 0.16\,ms.) \approx 8\,\text{sec}.$

#### Predictive store

The last option is the "predictive store" approach. Here the event of making a request for guidance, as described in the second option, is used. When a request for guidance is made, the central server "floods" the signs, which are on the route from the source to the destination, with sign specific direction information. After receiving the direction information, the sign store the information on a local repository. When a person passes by, the sign checks the internal repository and shows the directional information. The information will be stored until or a timeout is reached or the end of a path is reached. This option tries to use the smart signs to the fullest and can dynamically use more or less memory of the sign. Therefore the storage space for this option is the maximum available space.

Applied to the example situation, the following characteristics are calculated:

Storage space $= 200\,bytes = 200 \div 3\,bytes = 66\,guide\,items$ (max)
Smart Sign communication messages (while flooding) $= 1 \times 4\,hops = 4\,messages$

Reaction time (off-track) $= 8\,messages \times 1\text{sec.} + 2 \times (4\,ms. + 0.16\,ms.) \approx 8\text{sec.}$

Reaction time (after flooding) $= 0\text{sec.}$ (no communication)

**Summarising quantities**

| Aspect | Full store | No store | Predictive store |
|---|---|---|---|
| Storage space in bytes | 200 (max) | 21 | 200 (dynamic) |
| Number of communication messages | 4 on beforehand | 8 | 4, while flooding; 0, after flooding |
| Reaction time in seconds | 0 | 8 | 8, off-track; 0, after flooding |

*Table 4 : Summary calculated example situation*

**Chosen**

The selected choice is the "predictive store" approach, because it is best suited for the prescribed purpose. The other options are not desirable.

The "full store" is not the best-suited solution for this purpose. Although the reaction time is fast (in the example 0 seconds), the memory is constantly occupied with routing information. This leaves less space available for guidance information and for other functionalities, like showing information. Furthermore, the whole network stores the guide information for a particular user. In a storage scarce environment the superfluous information, which completely off-track smart signs store, is not desirable. This approach also does not react well to changes in the physical distribution and orientation of smart signs. All the signs need to be updated individually, before the changed physical representation can be used. Finally this option needs an initial round of message sending, for distributing the route information to every smart sign. The other options do not need this round.

The "no store" approach is doable, but not the best suited solution, because every time the guidance data needs to be fetched. This generates a lot of communication traffic back and forward between the signs and the central server. The figures stated in table 4 show that there are 8 messages needed to fetch information for one smart sign. Even for a small guidance the number of messages over the network will be a lot. Most power is consumed by the smart signs, while sending a message. With the big amount of messages, the power consumption will also be a lot. This power drain is not desirable, because this resource is scarce. Also a lot of messages will be superfluous, because they contain exactly the same information. All these messages are also not good for the reaction time, 8 seconds in the example, which is too slow.

The "predictive store" however generates a lot less traffic; only the flooding is expensive (max. of 4 messages per smart sign). Although the traffic is more spread in the "no store" approach, the "no store" uses the communication channel all the time and generates a lot of double and overhead communication. The flooding floods the network once and then is finished. When the flooding is done at a strategic time (before the user is actually guided), then it certainly is an advantage. The flooding also has the advantage, that it has a fast response time, when flooded, and still does not permanently occupy a lot of memory. One can see this back in the example situation. When the flooding is done, the reaction time is 0 seconds and dynamically uses the maximum amount of storage. The only problem with the "predictive store" is that when a person goes off the prescribed track, the directional information is not available. In that case the "no store" approach (with storage of recent received directions) is the only option to chose. Then the reaction time is 8 seconds. However, this combination saves space compared to the "full store", which stores the direction on every smart sign.

Summarising the "predictive store", as default approach in combination with the "no store" approach, is the best option for this purpose.

### 6.1.6 Printing

Printing shares some points with guiding. This is because, when a print request is made, the user has to be guided to a designated printer. Also the print request needs to be processed before / when the user arrives at the printer. The guiding part of the printing is the same as the guide case. The only thing that is different is the printing mechanism itself. There are several design options for this print mechanism.

### Direct printing

A document can be directly printed when a print command is given. This option has some major disadvantages. The first disadvantage is that when a print command is given, the printed document has to be retrieved from one particular printer. Another disadvantage is that the documents are printed directly, this can result into document chaos when multiple documents are printed in a short time. The only advantage of the system, over the existing way of printing, is that it automatically selects the nearest printer to the user.

### Zone printing

The second option is to print the document, when the person is close to a printer. Here some kind of zone around the printer is defined. When a person enters this zone, the document starts printing. This has the advantage that the trigger mechanism, of printing a document, is linked to the position of a user. Now a document can be printed on any printer at any time. The disadvantage of this mechanism is that, when a person enters the zone, the document will be printed and the user has to fetch his prints at that particular printer. If the user decides to go to somewhere else, the mechanism fails. This disadvantage can be softened by intelligently defining the zone around a printer.

### Behavioural printing

The final option looks a lot like the second option, but now printing is not initiated by entering a zone, but by recognising the behaviour of the user and attaching actions to certain kinds of behaviour. This behaviour recognition can be done in many ways and can have various degrees of intelligence. This mechanism can be done in such an intelligent way, that the printing is commenced, when a user also intends to get the print outs of a document. A simple example is calculating the direction a user is heading and undertaking actions accordingly. The disadvantage of this mechanism is that it can get very complex and that it needs precise information.

### Chosen

The final option, which links the behaviour of a user to actions, looks the best option. But this option is not the best-suited option yet, because the location information is not precise enough to use it as a consistent platform. Also the implementation can become very complex and is only a layer on top of the location information. This option is not feasible in the given time frame and platform. The static approach, on the other hand, is too static and does not use the present platform to its full extend. Therefore the second option is selected. The second option uses the location information and is able to show intelligent behaviour, without being too complex.

## 6.1.7 Showing Information

Information needs to be displayed on signs, which represent rooms or other environmental object. But also personalised information about these objects should be shown. The problems are in what way the information is stored and how information is shown to a user. How the information is shown is already discussed. Here the different solutions for storing information are discussed.

### Full Store

The first option stores all the information (both general and personal information) on the sign itself. The advantage is that everything is handled locally and thus has a fast reaction time. The disadvantages are that a lot of memory is needed especially when a lot of users are using the system (even when information is stored on the EEPROM). Also the management of the data is hard to do. Each node has to be accessed and changed to update information.

Applied to the example situation the following quantities are for the “full store” option:

Reaction time $= 0\,\mathrm{sec}.$ (no communication)

Needed storage $= 50\,users \times 10\,bytes = 500\,bytes$

## No store

The second option looks a lot like the “no store” option of the guidance case. Now all information is stored like the “no store” option of the guidance case. Now all information is stored on one central place and information is send to the node, whenever needed. The advantages and disadvantages are already discussed in the guidance case.

Applied to the example situation the following quantities are for the “no store” option.

Reaction time $= 2 \times ((4\,hops \times 1\sec.) + 8\,ms. + 0.32\,ms.) \approx 8\sec.$

Needed storage $3\,displaycomponents \times 10\,bytes = 30\,bytes$

## Partial store

The third option is to use “partial store”. Here the static data is stored permanently on the sign and the dynamic data is stored on a central place and can be send on request. This dynamic information is stored for a period of time (cache) on the sign. The static data is the general information and the dynamic the personal data. The advantages are that the dynamic data is managed centrally and can handle the scarce memory, by throwing away data when it is old or when the memory is full. The disadvantage is that the reaction time is slower than the “full store” for the dynamic data. However, the data is cached for later use.

Applied to the example situation the following quantities are for the “partial store” option.

Reaction time (information not available) = “no store” $\approx 8\sec.$

Reaction time (information available) = “full store” $= 0\sec.$

Needed storage = maximum available memory $= 200\,bytes = 200/10 = 20\,items$

## Predictive store

The final option is that of a “predictive store”. Here the dynamic information is send to a sign, when the intention of a user is to pass by that sign. When this intention is sensed in an early stage, then the response time will be like the “full store”, but the information can be managed centrally. Still the disadvantage of all the communication messages between signs stays. Another disadvantage of this approach is that a prediction algorithm needs to be developed. This can become complex and needs a solid localisation platform.

Applied to the example situation the quantities are the same, except for the reaction time. A message takes 8 seconds to be shown. When the intention of a user is predicted well, it finds it out more than 8 seconds before arrival. This makes the reaction time 0 seconds.

## Summarising quantities

| Aspect | Full store | No store | Partial store | Predictive store |
|---|---|---|---|---|
| Needed storage in bytes | 500 | 30 | 200 | 200 |
| Reaction time in seconds | 0 | 8 | 8, when not available; 0, when available | 0 |

*Table 5 : Summary calculated example situation*

**Chosen**

The “full store” approach is not feasible, because storing all room information, including the personalised information, requires too much of memory (especially when having a lot of users), which does not comply with the memory lean smart signs. One can see this in the example situation. It uses 500 bytes only for this section of the system, while the memory is only 200 bytes. This can become a big problem in big offices. In the example situation only 50 users are included, eventually more users are going to use the system. One can see that the amount of used storage space (500 bytes) already exceeds the memory capacity (200 bytes). Also updating the very dynamic personal information is tough.

The “no store” approach is too lean. Every time somebody passes by, all the data needs to be fetched. This generates a lot of traffic on the communication channel (example: 8 messages per sign, per fetch). This communication soaks up the energy from the smart signs. Although the personal information is quite variable, there will be too much superfluous and double information send to the smart signs. Also the reaction time is too slow, it takes around 8 seconds, in the example, to show information. That is too long.

The “partial store” however is a good meet in the middle (between “no store” and “full store”) approach. Here the general information is stored and also the latest personal information. The storage of the personal information is also so variable and can become that many, that it can not be fully stored on a sign. Now the maximum storage is maximally used, without exceeding the maximum capacity (in the example 200 bytes). The cache is the only reasonable solution to try to store some of the personal information regarding that sign. The disadvantage of having a slow response time, when the information is not available in the cache, can be neutralised by flooding all the smart signs in the network with relevant information. If this flooding is done at strategic times, the personal information will be available in the cache of every smart sign and thus have a fast response time.

Using “predictive store” is also a good option. In the example it scored ideally: reaction time 0 seconds and dynamically using the maximum amount of storage (200 bytes). However, this option is too much overkill in this context. The algorithm can become very complex and the location platform is not accurate enough. This option can maybe be chosen in the future, when a more solid location platform is available and when the partial store does not satisfy the needs.

Summarising, the “partial store” is the best-suited option for this context.

# 7 Architecture

After deciding on the important design decisions. One has a vague view of what the system should look like. Here the mapping of the design decisions to the architecture will be discussed. First the system will be described on a high level. Thereafter the level will descend to a lower level, where the architecture will be described in more detail.

## 7.1 Initial architecture

The design choices result into a combined central and distributed approach. Here all information is stored centrally and a cache is stored on the smart signs. There also needs to be a way to approximate the location of a user, to be able to guide a user and to use “zone printing”. Finally a user has to be able to place a request for guidance, or to print documents.

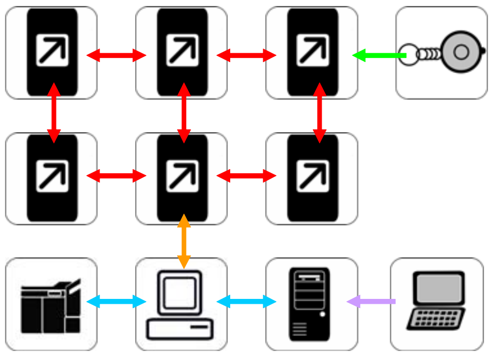


Figure 6: Schematic architecture

To be able to have this combined central and distributed approach, there is a need for a central server, which can process difficult tasks and stores all the information. There is also need for communication between the central server and the smart signs network. Having a gateway does this. The gateway has two sides: smart sign side and computer side. Also a tag needs to be present in the architecture, to get an approximation of the user location. Finally, to be able to have user interaction, a request computer needs to be included in the architecture. The design choices furthermore describe the usage of a display and printer. This brings forward the high level architecture of the Smart Signs system, which is described below. Figure 6 already shows the schematic representation of the high level architecture.

### 7.1.1 High level architecture

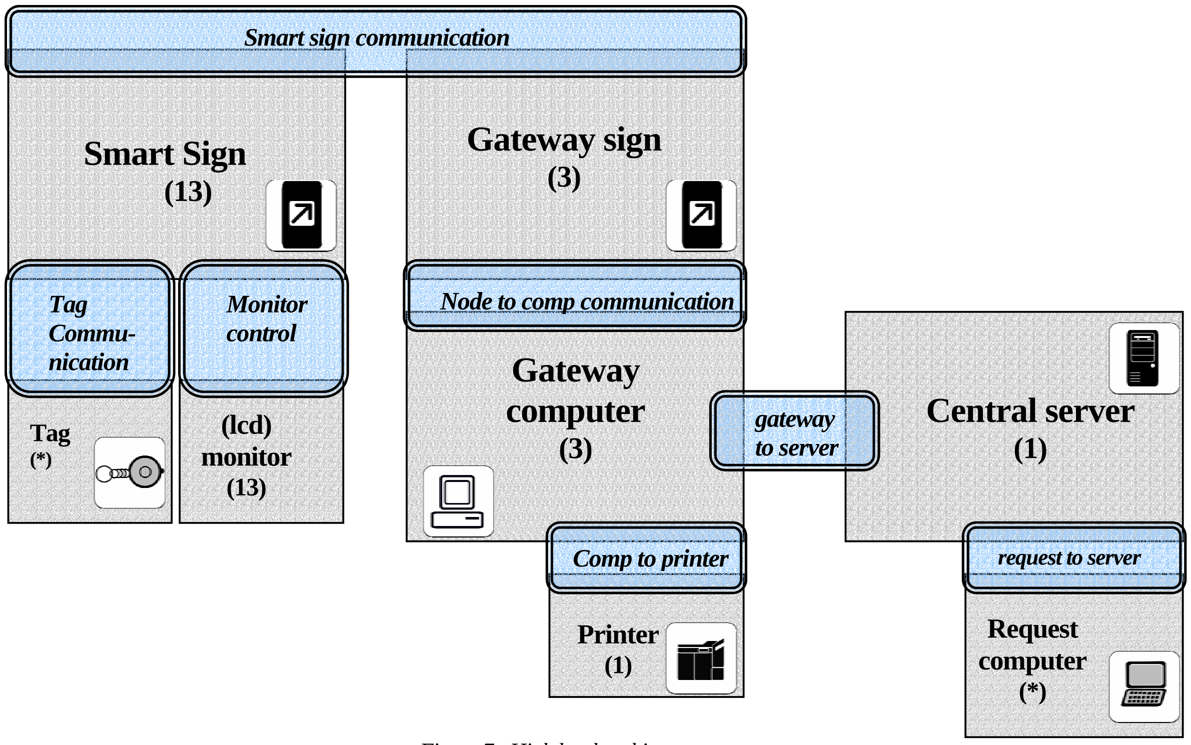


Figure 7 : High level architecture

The numbers between the brackets are the quantities (* = variable), used in the SVG conference (see chapter 9), to give an idea about the proportions between the components.

Hereunder the highest level entities with their communication channels are briefly deliberated.

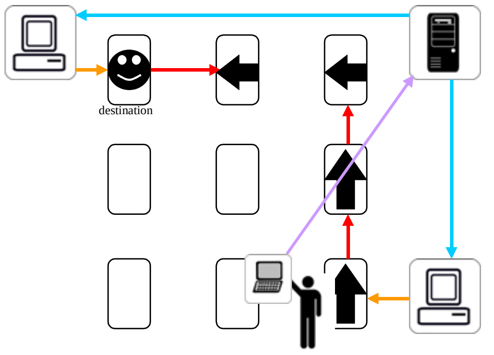
- _Central server:_ This component is the heart of the Smart Signs system. It contains the topology of the Smart Signs network, all the node information and all the print information. This information can be send on demand to a smart sign, through the gateway computer. Also directions are generated, while processing a guide request. This guide request comes from a request computer.
- _Gateway computer:_ Computer side of the gateway, which exchanges data between the central server and the smart signs network. It passes node, print and guide information from the central server to the gateway node, where the information is further processed. Also the event of a newly sensed tag is passed on from the gateway node to the central server. The gateway computer can furthermore be linked to a printer to be able to print on demand of the central server.
- _Gateway Sign:_ Smart Sign side of the gateway, which passes messages from the Smart Signs network to the computer side of the gateway. It also contains the functionalities of a Smart Sign.
- _Smart Sign:_ A component of the Smart Signs network, which for instance locally manages cached information. The Smart Signs are able to pass messages to each other.
- _Tag:_ The component, which a user carries around, sending out a unique identifier, which a smart sign can intercept.
- _Monitor:_ Where the information is actually shown to a user, controlled by a Smart Sign.
- _Printer:_ A printer, who prints documents, initiated by the gateway computer.
- _Request computer:_ A computer on which a user places a request for guidance or a print job.

More detailed architectural information about the internal usage of the system will be given after the deliberation of the flow of communication.

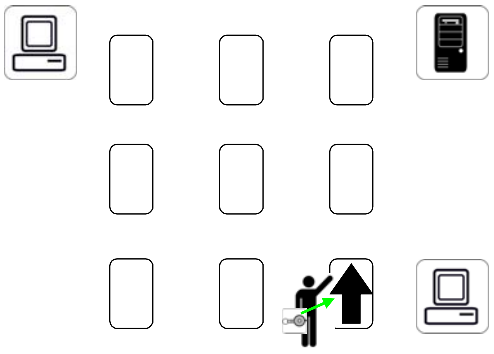
### 7.1.2 High level communication flow

To describe the flow of communication between the high level components described above, a variant on the sequence diagram is used. This gives a good descriptive and graphical definition of the high level communication. The sequence diagrams enclosed in appendix G give more detailed information.

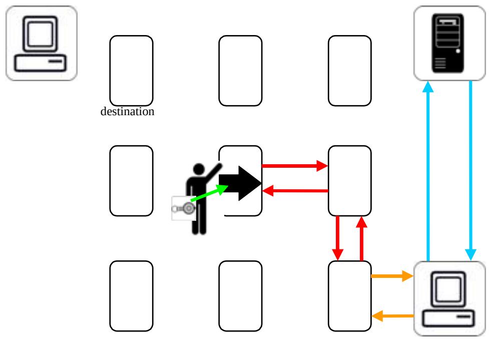
**GUIDING**



1. When a guide request is made, the smart signs, which are on the path from a source to a destination, are flooded with directional information. The server gets a request from a request computer. The central server sends directional information to the smart signs, on the path to the destination, using the gateways.

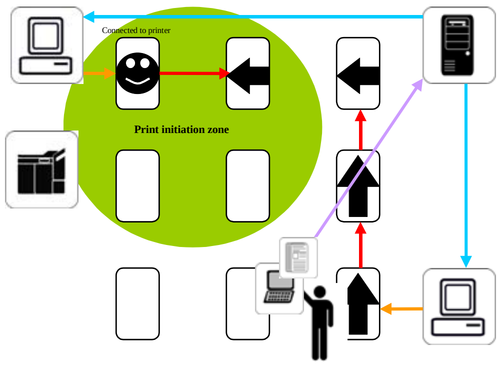


2. When a user passes a smart sign on the path, then the information is already present at the node and thus directly shown.

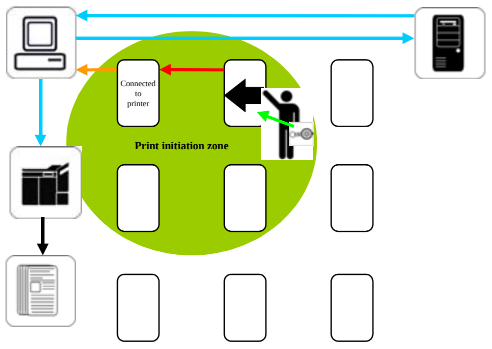


3. The case of the user going off path results into the smart signs, which senses the user, sending status information. The central server checks this status information and sends missing information to that particular smart sign. In this case also directional information is send to that sign, which the sign will show immediately, when the received information is applicable to a user.

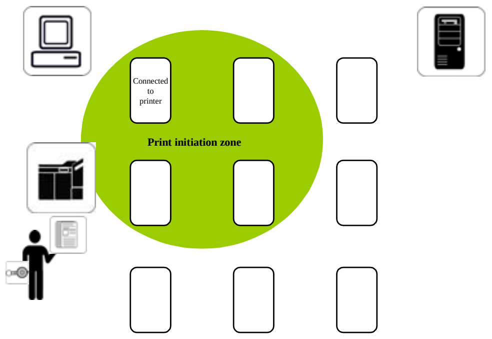
## PRINTING



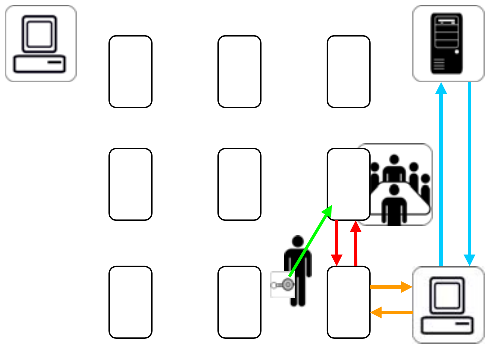
4. When a user prints something on a request computer a connection is made to the central server, where the documents are physically stored. Also a guide request is made automatically and the user is guided to the nearest printer.



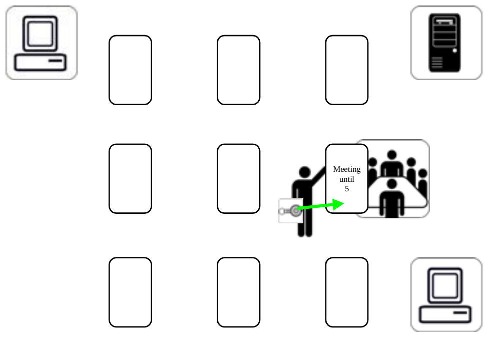
5. A print command is given, when a user is in a particular zone around a printer and the user has pending prints. A smart sign first senses the user. This smart sign sends status information about the newly sensed user. Then the central server checks whether anything needs to be printed for that user. When the user has prints, then a print command is given to the printer, which the user is close to.



6. When the user arrives at the printer, the documents are already printed and ready to be taken.

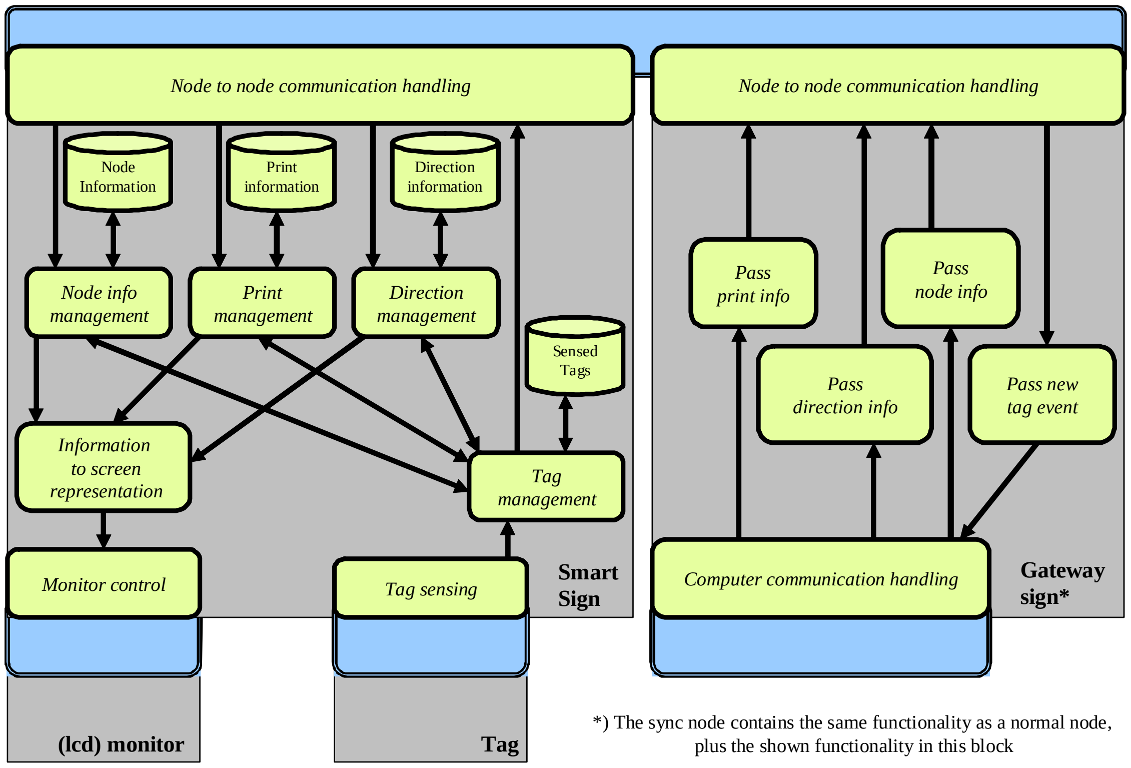


7. Personalised (object) information is shown, when there is information available at the cache of that particular smart sign, or when the central server has information for the user. In this case the smart sign does not have cache information for a user and senses a tag. Status information about that tag (user) is send to the central server (status about the print, node and guide information). The central server also checks for personal object information, which it has in this case. The central server then sends the found information to that particular sign.



8. In the case there is cache information, or when after a little while the user stands in front of the sign, the information is available right away. Now only communication from the tag to the smart sign is needed.

### 7.1.3 Low level Data Flow Diagram



*) The sync node contains the same functionality as a normal node, plus the shown functionality in this block

*Figure 8: The smart signs network*

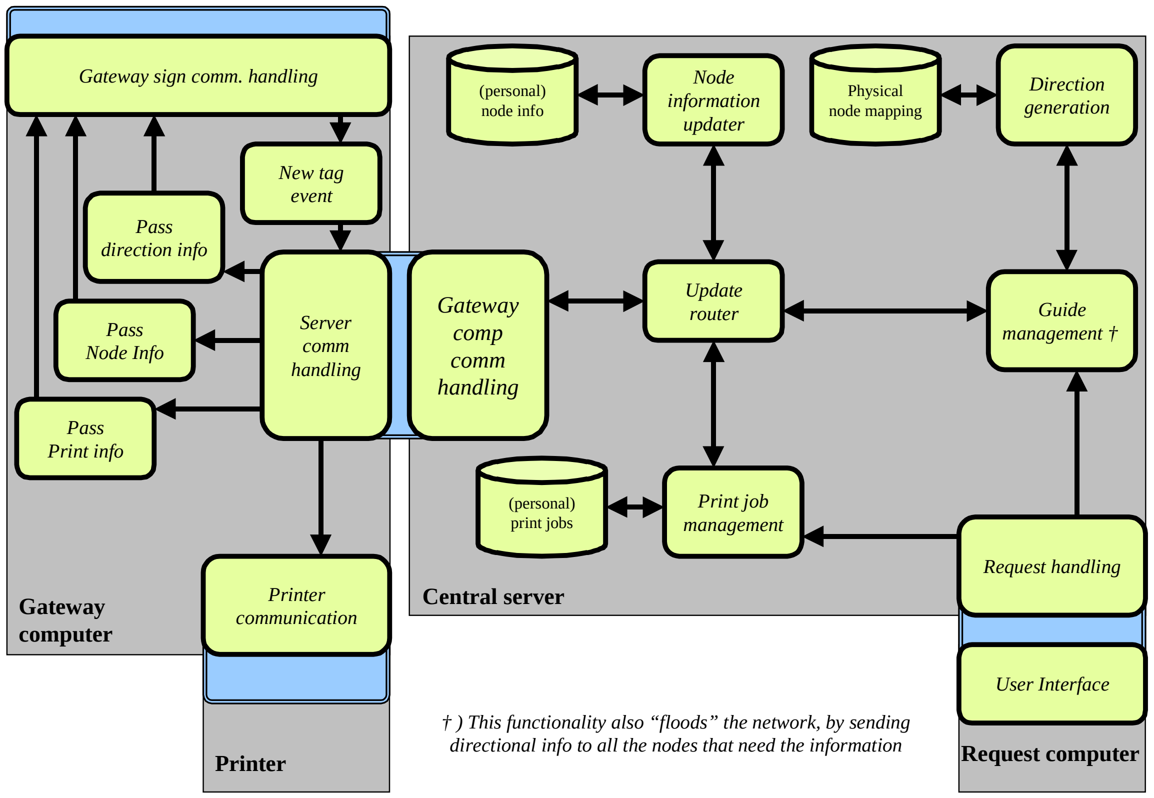


*Figure 9: The computer side*

To be able to understand the detailed architecture, one has to descent one level in the architecture. Appendix E shows the detailed description of the entities and their relations concerning Figure 8 and Figure 9. Here a short description of the to be performed functions of the high level entities are described, which serve as a basis for the low level architecture:

1. *Smart sign*
   1.1. Sensing of Tags and keeping track of tags in the proximity of the smart sign.
   1.2. Storing a cache of (personal) node (object), print and direction information.
   1.3. Show information on a connected display.
   1.4. Send an event that a new tag has been sensed, through a gateway sign, to the central server, where it is checked whether information should be send to the smart sign.
2. *Gateway sign*
   2.1. Perform the same functionalities as a smart sign.
   2.2. Pass node (object), print and directional information from a connected gateway computer to the smart signs network.
   2.3. Pass the event that a new tag has been sensed by one of the smart signs to a gateway computer.
3. *Gateway computer*
   3.1. Pass node (object), print and directional information from a central server to a gateway sign, where the information can be send to a particular smart sign.
   3.2. Give print commands to a connected printer.
4. *Central server*
   4.1. Store all information (node and print information).
   4.2. Produce directional information.
   4.3. Handle guide requests and print commands from a user.
   4.4. Manage to be printed documents.

## 7.2 Final architecture

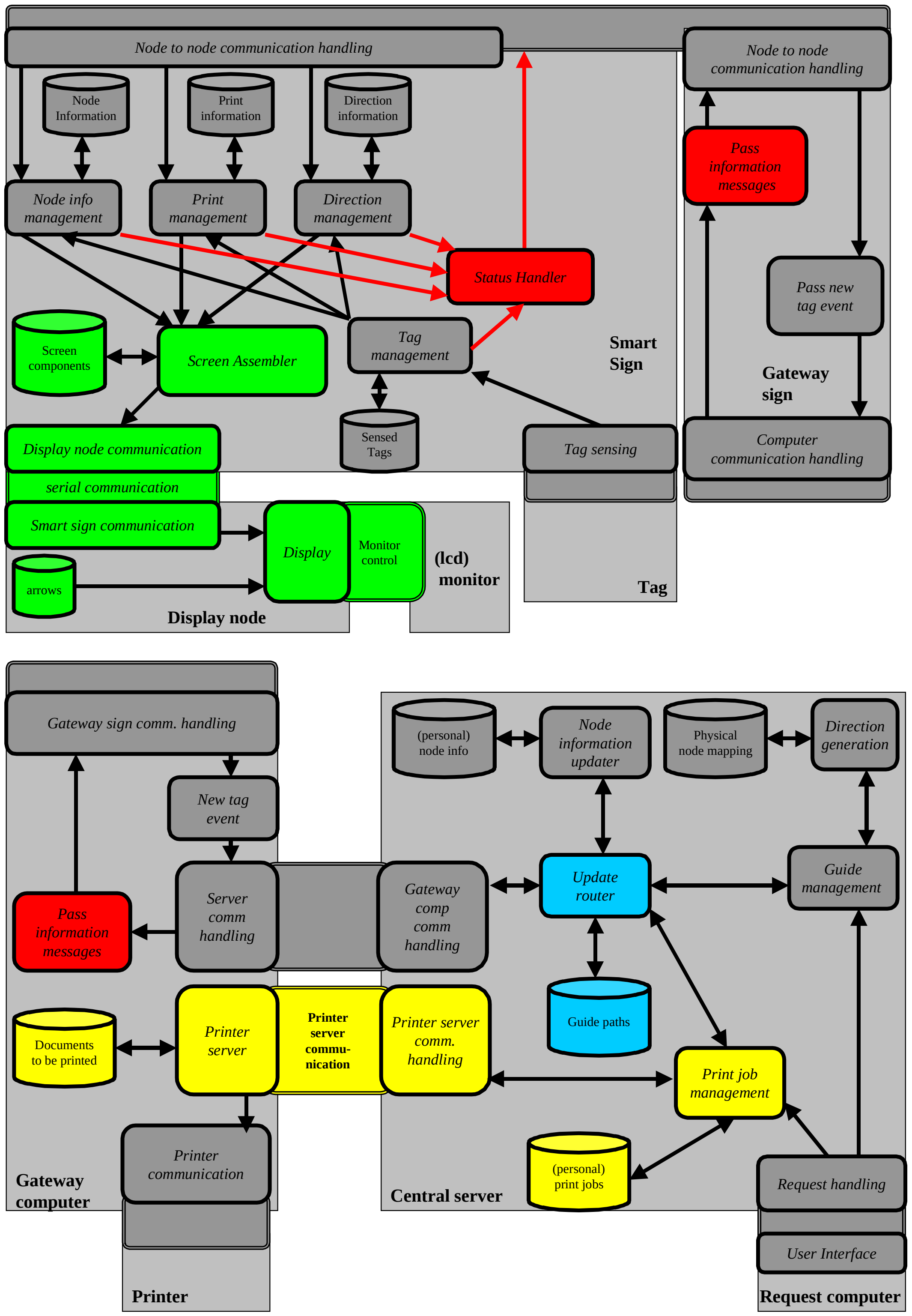


*Figure 10: Final detailed architecture*

The initial architecture turned out to be a good basis for the implementation. However, some adjustments are made, to make the design more feasible for the actual implementation. Here the changes are described, which lead to the final architecture. The Data Flow Diagram of the final architecture can be seen at Figure 10 and appendix F.

### 7.2.1 Monitor extension

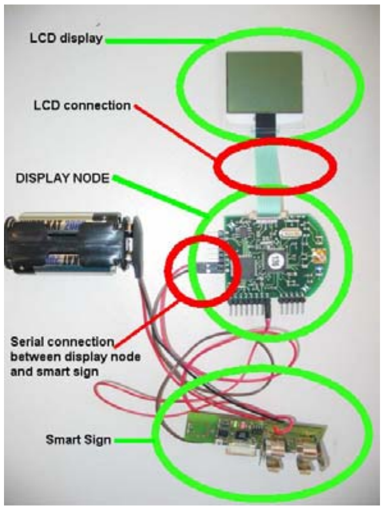


*Figure 11 : bare Smart Sign*

In the final architecture the monitor is not a passive component anymore. The reason behind this is that for the final implementation (Figure 4, the latest smart sign), used for the SVG conference demonstration, the smart sign does not have a display connection. It only has a serial and wireless connection to communicate with other components. The only way to have a display for this smart sign is a serial connection to another node ( the display node), that receives information from the smart sign. This information is then shown on the LCD display of the display node. The display node includes the following functionalities:

- Receiving (serial) information packages
- Digesting information packages
- Storing directional arrows
- Show node, print and guide information in the right way on the correct place.

In Figure 10 this change can schematically be seen, marked with the colour green. The picture on the left shows the real set-up.

### 7.2.2 Path tracking

In the initial design there was no tracking of which users are currently guided to where. This facility is needed in the central server, to cover the off-path scenario of the guide case. When a newly sensed tag message arrives at the central server, then it should be possible to check whether the user needs directional information (when the status message indicates no present guide information) and what directional information the user needs. In figure 10 this change is marked in blue.

### 7.2.3 Printer server

With the SVG conference in mind and the time pressure, storing all prints on the central server turned out to be not feasible. Also the already present solution for managing prints was good enough for a basis. This already present solution needs the windows operating system, whereas linux is used at the central server. These facts result in the migration of print management to one of the gateways. Now all the prints are stored at a gateway and a separate connection to the central server is present, for the initiation of printing. The changes are marked in yellow at Figure 10.

### 7.2.4 Other small changes

Apart from the above-described changes, some small changes were made. These changes mostly involve shifting functionality from one process to another, within a component. All the small changes are marked with the colour red in the Data Flow Diagram of the final architecture, seen at Figure 10. Hereunder the small changes are shortly described:

- Passing information (gateway): Putting together the processes, which pass node, print and direction information is more simple and efficient. This is because here the system only has to pass the information on and does not need to know what kind of data it is.
- Passing information (sync node): see first point.

Status gathering (smart sign): Gathering the status of whether the information is present or not needs a process for itself. This is because the status information comes from different processes and can only be send when all information is there. The other components only manage one kind of information and can therefore not have the overview.

## 7.3 Example network

Here one can see the network set-up of the SVG conference, as an example to show how a Smart Signs system can look like. For the physical representation of the network, please see Figure 18.

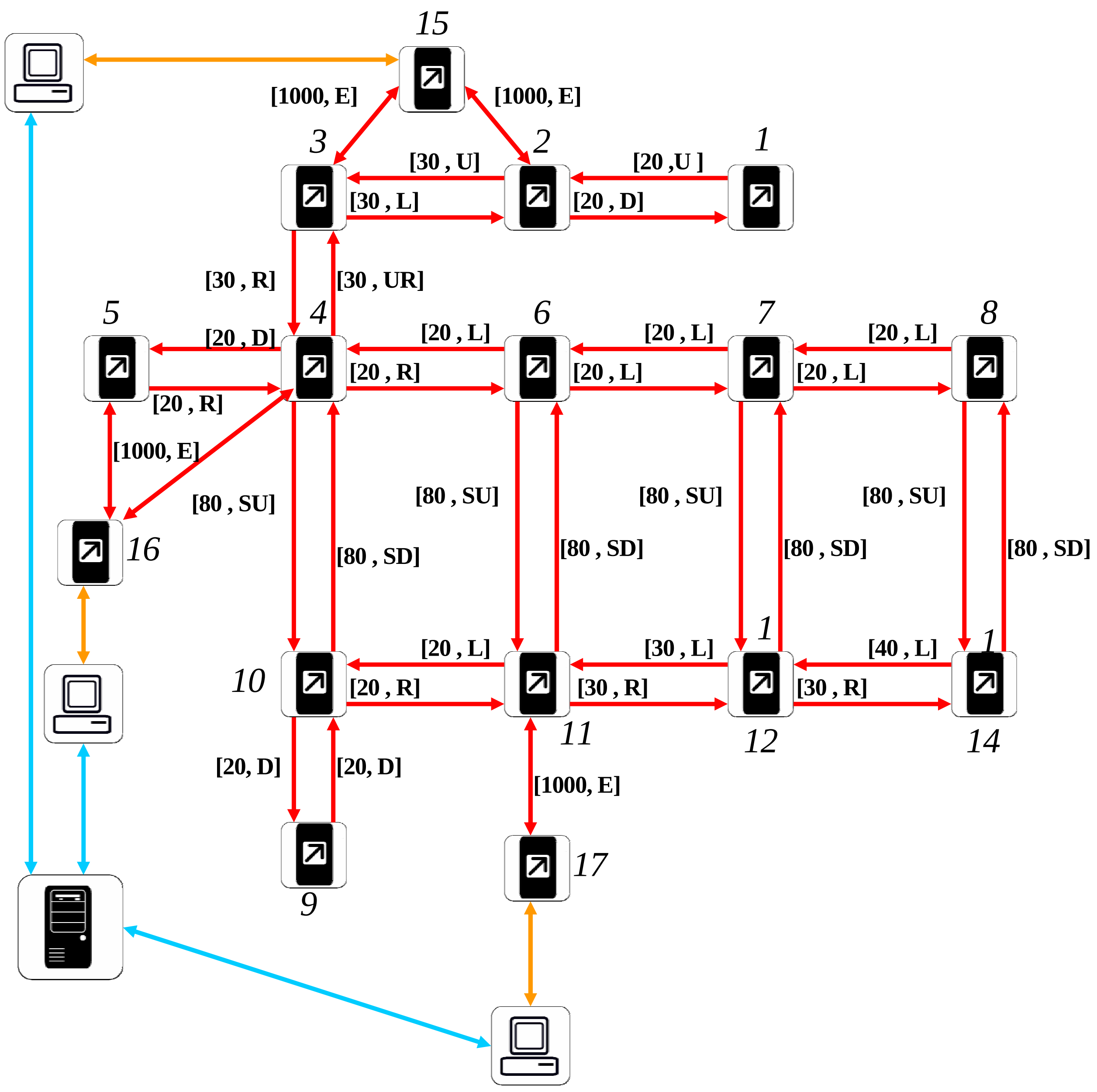


*Figure 12: SVG network structure*

*Figure notes:*

- The pairs connected to the arrows represent: [weight (approximate length), arrow the sign should show (see Figure XXX)]
- Number 13 is not present in the network, because there were not enough smart signs. Therefore one had to be removed.
- Going up and down the stairs have bigger weights. This is to make sure that a person has to switch floors only once, while guided. Otherwise one has to climb the stairs all the time.
- The connection between a gateway sign and another sign always has the weight 1000. This is because in this set-up the gateway signs are not included for guidance or displaying information, just to act as a gateway. So one does not want to be guided to there.

# 8 Implementation

This chapter discusses per component (of the final architecture) how it globally was implemented and what was used for implementation. Also is discussed which obstacles occurred while implementing and how these were solved. The implementation process is only described globally. For details, please read the documentation of the code, found at appendix I (enclosed CD). The implementation is mainly described using the low-level architecture, mentioned in chapter 7, section “Final architecture”.

## 8.1 Central server

### 8.1.1 Actual implementation

*Figure 13: Central Server*

The implementation of the central server is done in the environment of Linux, because of the familiarity with Linux in combination with databases, web sites and sockets. The central server could be implemented as a part of a gateway. However the decision was made to keep the components on different platforms, to be able to manage the central server apart from the gateways. It also gives a clear separation of the different functionalities of the system. Hereunder the actual implementation of the functionalities is discussed.

#### Store all information

For storing the node and print information a MySQL database is used. The MySQL database is used, because MySQL has a simple interface and can be managed through a web site. This has the advantage of having a good overview and being able to manage (delete, add, modify) information easily. This option is also a good basis for further development, because the information is now externally stored and can thus easily migrate to other storage platforms. For more details about the database structure and what is stored, please see appendix H.

#### Produce directional information

The production of directional information can only be done, when the central server has awareness about the physical distribution of the network of smart signs. Only then the central server can give sensible guide information. The central server needs to be aware of the following things, regarding guiding persons:

- The (relative) location of a smart sign.
- The directions to go from one sign to another.
- Whether a sign is connected to a gateway and/or a printer.
- The distances between the smart signs.
- The shortest path between two signs.

To get the awareness of all these points a weighted directed graph is used, with the storage of extra node (gateway, printer) and edge (direction) information. To get the shortest path from one sign to another, Dijkstra’s algorithm is used [DIJK]. It calculates the shortest path, using the weights of the edges between the signs.

Also a mechanism for checking whether a user is being guided is needed, to check for available guide information for a particular user. This is done by storing the guide information in a database, where the central server can check whether the user is being guided and whereto. For more information, please see appendix H.

#### Handle guide requests and print commands from a user

Handling guide requests and print commands are both done in a different way. Handling guide requests is done using sockets, where users can connect to. The choice to use sockets is made, because sockets are universally used in network environments and enable many different types of interfaces, which are build on top of the socket mechanism. One of these is a web interface. This kind of interfaces are commonly used and therefore a solid choice to use.

Handling the print commands are done through a different channel. Here the normal path of printing is used. Only now a printer is selected, which internally forwards the documents to the printer server and stores it on the gateway, to be initiated at a later time. For more detailed information, please see Figure 10 and appendix H. This part of the implementation deviates a little bit from the final architecture. There the central server does the communication and here the gateway arranges the handling of print commands.

#### Manage to be printed documents

Although the documents are not physically stored on the central server (see chapter 7), a list of job identifiers, linked to numbers of actual prints jobs stored on a gateway, is stored along with other user information. With this mechanism the central server can keep track of the to be printed documents. The central server can furthermore initiate the printing by sending a command with a list of print jobs, which are then actually printed using the gateway computer.

### 8.1.2 Obstacles and difficulties

This component was not the hardest to develop. This is because of the familiarity with the programming environment. The only real implementation obstacle was finding a way to create awareness of the central server about the physical distribution of the network of smart signs. How this was tackled is described above.

## 8.2 Gateway computer

### 8.2.1 Actual implementation

The gateway computer is implemented in Windows, using C, because the gateway program has to run in the background of an end-user computer, where people for example can print or place guide requests. This kind of machines is generally equipped with Windows. The programming language C is used, because of the ability to easily realise low level communication, like the communication through a serial port. The communication between the gateway and the gateway sign is done though a serial port [SCW]. Hereunder a global explanation of the functionalities, given by the design, is given. For more detailed implementation information, please see appendix I (enclosed CD).

*Figure 14 : Gateway computer*

#### Pass information from a central server to a gateway sign

Passing is done by forwarding all messages to the connected gateway sign. The type of the message is checked on reception from the central server (socket), whether the message contains information data. When this is the case the whole message is send to the serial port, where a gateway sign is connected.

#### Give print commands to a connected printer

The actual printing of a document is done in co-operation with the central server. The central server gives the gateway a set of job identifiers, which have to be printed. The gateway computer then sends the resembling stored files to a designated printer. The files are stored on the gateway computer, because a print command given by a user to a “fake” printer are handled by the gateway computer. A stub printer is used, which sends the documents, which are given by the user through the print command, to a file. These files can be printed later, as described above.

### 8.2.2 Obstacles and difficulties

The gateway was also not a big problem to implement, although the experience with programming in Windows was not that much as in Linux. There were some obstacles/difficulties while implementing. The biggest obstacle was getting the serial communication right. The communication is bi-directional and thus a read and write can be done simultaneous. It was quite a hassle to get this done in Windows. Another obstacle was getting the information passing right from the central server to the sync node and vice versa. This is done parallel and thus needs threads. Getting this to work together with the serial communication was a bit of puzzling, because of the lack of documentation.

## 8.3 Smart sign, gateway sign and display node

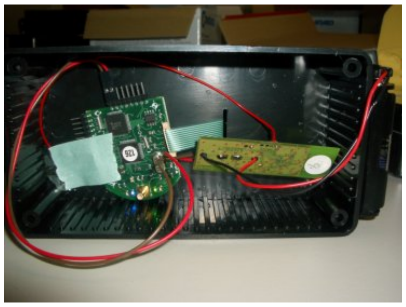


*Figure 15 : smart sign (inside)*

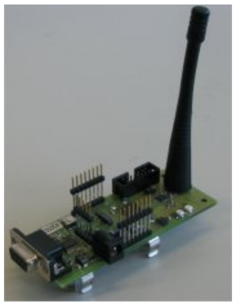


*Figure 16 :gateway sign*

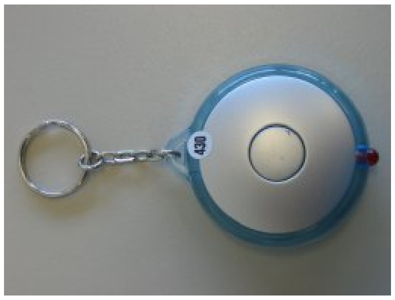


*Figure 17 : Tag(keychain)*

### 8.3.1 Actual implementation

The Smart signs, sync and display nodes are implemented using the given platform (see chapter 5). The platform requires to program in a variant of C and using the AmbientRT operating system. The implementation is done in modules, just like the central server, to keep a link to the design and to keep overview. How the functionalities, described in chapter 7, are implemented is described below. For more detailed information about the implementation, please read the documentation of the code, enclosed in appendix I (CD).

#### Sensing of tags and keeping track of tags in the proximity of the smart sign

In the final implementation the sensing of tags is included in the transport protocol  (mac protocol) of the operating system. In earlier versions this was done on top of the transport protocol (lmac protocol [LMAC]). This is included in the transport protocol, because the final version of the tag does not have the resources to use the full transport protocol and thus has a different protocol. This protocol needed to be embedded in the transport protocol.

Keeping track of sensed tags is done by storing the tag information in a dynamic linked list. The linked list is a feature of the operating system. The tag information includes the identifier and its age. The age of the tag is incremented each second and is used to keep track of how long ago the last time the tag was sensed by the smart sign. If the age becomes to big, then the tag (and thus the user) is no longer in the proximity of the smart sign.

#### Storing a cache of (personal) information

For storage of information a dynamic linked list is used, provided by the operating system (as described above). The linked lists are used, because they are easy to manage and the framework for dynamically storing information is already present.

#### Show information on a connected display

The information, which is stored in dynamic linked lists, is displayed though a display node. The smart sign first assembles all the relevant information. This is done by mapping the sensed tags to the present information. All relevant information, which can be displayed, is stored in a screen components linked list. Then it is decided which information should be displayed, some information has a higher priority. This information, along with positional information, is then send to the connected display node. Here the display node unpacks the information and displays the information on the screen.

#### Send a new tag sensed event, through a gateway sign, to the central server

A smart sign sends a status message of a newly sensed tag, because then the central server can look for information meant for that smart sign. When a tag is newly sensed (there was no present tag information in the linked list of tags), a signal is send to collect status information about the existence of information for that tag. Then the information is send to the nearest gateway sign.

#### Pass information from a connected gateway computer to the smart signs network

Forwarding information is only done by the gateway sign. The implementation is done by checking the incoming message for its type and destination. If the destination is not the gateway sign itself, then it is forwarded to another smart sign, for further routing. The routing is done by an array of smart sign identifiers, maximum hops big. This array is filled by the central server, which knows how to route the message to the destination. In this array smart signs can see whether the information is for the smart sign, or whether it should be forwarded and to which smart sign.

#### Pass new tag event to a gateway computer

This functionality is also only for the gateway sign. This is implemented in quite a straightforward manner. The gateway sign receives a “newly sensed tag” status message (described above) and directly writes it to the serial port, where a gateway computer is connected.

### 8.3.2 Obstacles and difficulties

These components turned out to be the most difficult components to implement. There were several difficulties, which had to be overcome.

#### Unknown platform

The platform was completely unknown. In combination with the fact that the operating system differs a lot from the most commonly used operating systems, it was quite an obstacle to overcome. The environment first had to be investigated. The first acquaintance was done at the University of Twente (see appendix A and B for the planning). Thereafter most of the environment was known, but still while implementing the components, there were some vague problems, that needed help from the creator of the operating system. Most obstacles were because some things were unclear. Also the structure of the operating system needed some time to get used to. Some mechanisms were very different. For example defining the different tasks (for more details, please see [ART]).

#### Hardware boundaries

The hardware boundaries really had to be taken into account. The memory, for example, became a big obstacle at a certain point of time. The memory became full real fast. When data was moved from the main memory to the flash memory, some pressure drained from the main memory, but still some information still needs to be stored in the memory, like pointers to the position where the information is stored. This results in the fact that not a lot of information can be stored. Also the processor has limited capacity, which translates in keeping the application as simple as possible.

#### Software

While implementing a subset of C and a limited amount of OS functions could be used. This sometimes resulted into finding a different implementation than usual. The used functions also look somewhat like their counterparts, used in commonly used operating systems, but then quite different. This was confusing in the beginning.

### Debugging

Debugging the implementation was quite difficult. First of all, a complete program needed to be compiled and flashed into the device, before the application could even be tested. While running the application there are several ways to get feedback from the system. The feedback can be done using LED’s, the serial communication port or the LCD display. The easiest way for debugging/feedback is the serial port. But when the serial port is already used for other communication, then the serial port can not be used for debugging. The LCD screen is not that good for debugging, because only a limited amount of information can be shown and displaying information is not done in a trivial way. The last resort is using the LED’s. This is only when the LCD is not present and the serial port is already taken. This scenario occurred a couple of times. The LED’s let you test only a small piece and can only give fail/pass information.

### Hardware changes

Some obstacles were related to changes enforced from outside. The hardware changes involved rewriting some functions to control hardware components on the smart signs. These changes had to be made, because the hardware that was used in the first instance (Figure 4, initial smart sign) turned out to be different than the hardware which was used for the prototype (Figure 4, latest smart sign) (at the conference). This change was done quite quickly, but another change in hardware had more impact on the system. One change in hardware resulted in a radical change in the system; even the design needed to be changed (see chapter 7, Final architecture). This change concerned the lack of a LCD display connection on the signs used for the prototype (at the conference). Because of the lack of this functionality the system needed to be cut into a display node and a smart sign. This is also described in chapter 7 (Final architecture).

### Software changes

Also the software platform (AmbientRT) had some changes that affected the implementation. This was mainly about the protocol for wireless communication between the smart signs. In the beginning of the project there was no protocol. After a while the LMAC protocol was used for communication. This resulted in changes for the smart signs and the LMAC tag. Just before the deadline of the prototype another change in protocol was made. This also resulted into changes. These changes were bigger, because the changes were done at a later stage of the implementation process. Also some software bugs resulted in extra work.

# 9 Practice

This chapter discusses testing and demonstrating the Smart Signs system at the SVG conference at the University of Twente. It first describes the environment, where the demo was held. Thereafter the performed tasks and the goals for multiple parties are discussed. Finally it is discussed what should happen and what actually happened.

## 9.1 Planned

For tackling the challenge to get the demonstration online a planning was made (see appendix B). As seen in the planning, one week was planned to get everything done for the demonstration at the first day of the conference. Hereunder the goals and tasks of the week of preparation are stated. Thereafter the environment, where the Smart Signs system had to be demonstrated, is briefly discussed.

### 9.1.1 Goals

The demonstration at the UT was entered with the following goals in mind:

- Show a working prototype of the Smart Signs system.
- Get user feedback of the usage of the prototype.

### 9.1.2 Tasks

While preparing the demonstration, the following tasks were assigned:

- Port the Smart Signs system, from the platform used for developing, to the platform used for the demonstration at the SVG conference.
- Test the ported system in the set-up of the demonstration.
- Arrange the infrastructure on which the Smart Signs system can run at the University of Twente (except the printer).
- Define a cut in the housing for the LCD connection.

### 9.1.3 Environment

The environment in which the Smart Signs had to be demonstrated, was the building “Waaijer” at the University of Twente. In this building, the smart signs had to direct people to one registration desk, three conference rooms and one elevator. Also one Océ printer was used to enable printing for the users of the system. In Figure 18 one can see the environment, where the system had to be demonstrated.

## 9.2 Actual

While the planned course of actions was straight forward, the actual execution of the plan turned out to be very different from that. Hereunder is first described how the porting and the set-up of the infrastructure were done. Thereafter it is discussed what problems occurred during these two activities. Finally it is discussed what went wrong while preparing the demonstration.

### 9.2.1 Porting

The main activity for the preparation was porting the Smart Signs system from one platform to another. This turned out to be a lot more work than anticipated before. On beforehand it was told that it was going to take one or two days. Although only two days (maximum) was required, a full week was planned mainly for porting. This was done to keep a buffer for unforeseen problems. This buffer turned out to be well needed and even not sufficient. The actual porting did not need to take a lot of time, but other problems popped up, which needed attention for getting the code to run on the platform used at the demonstration.

### 9.2.2 Infrastructure

Setting up the infrastructure involved two activities: setting up the smart signs in the environment and setting up the personal computers (gateway and central server). Setting up the smart signs was done by first selecting strategic positions for the smart signs. This involved communication with the co-organisers of the conference, to pinpoint the to be used facilities. The positions were tested for their range, by walking through the empty "Waaijer" and holding the signs and watch their status. The final distribution of the signs is shown below. Setting up the gateways and central server mainly involved installing the new gateways and getting the gateways and central server to work at the local network. For a more detailed picture of the infrastructure, please see Figure 12.

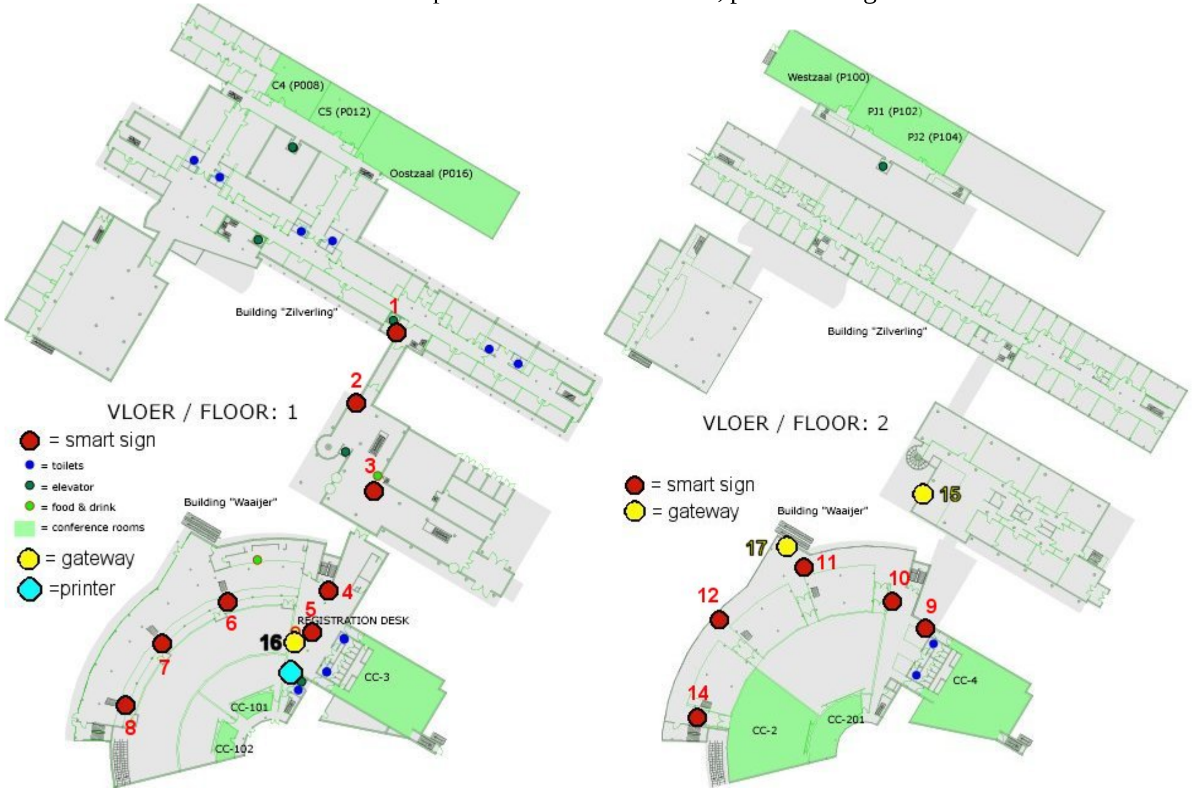


*Figure 18: Distribution of Smart Signs at SVG conference*

### 9.2.3 Problems

One of the problems that occurred was that some last minute changes were made in the platform. There were two big changes. The first one was newly needed protocol, which was finished only two days before the demonstration. This protocol was not thoroughly tested and still needed some adjustment. The change of protocol affected my code for the smart signs just a day before the demonstration. The second last minute change was a change of the communication between the smart sign and the display node. Although it was decided before, that the communication had to be done with acknowledgements, it became clear that it would not be possible to do the communication with acknowledgements. This resulted into quite some extra work and brought forward new problems regarding timing.

There were also unforeseen problems. These problems were regarding the depletion of the smart sign's main memory. This problem needed a lot of attention, because otherwise the smart sign would not work. Because of this problem, a rewrite had to be made, which uses the flash memory more. While solving this problem, some other problems linked to the flash memory occurred, which also needed attention. Also other unforeseen problems occurred while setting up the infrastructure. This turned out to be more work, than anticipated.

## 9.2.4 What went wrong

The demonstration turned out to be a failure. The Smart Signs system was not working when it was supposed to work. After moving the demonstration to a later day at the conference, the demonstration still failed. The reason behind this is elaborated below.

During porting and arranging the infrastructure the problems stated above occurred. These problems lead to the failure of the planning. All the time was consumed with porting the code and solving the problems. This left no time for testing the actual Smart Signs system. On the day of the demonstration the system was not tested at all in this configuration.

Another factor, for failing, was the used platform. Test results, from tests performed afterwards, indicate that the Smart Signs code works properly (apart from some minor bugs) and that the underlying software and hardware platform showed some weak point, which could have lead to the improper working of a smart sign. Some of these weak points are:

- Communication channels (serial and LCD): the channel between the smart sign and the display node and the display node and the LCD display sometimes stopped working.
- LCD: the display sometimes did not display anything at all, while the display node indicated that it should display something. It also sometimes stopped displaying all of a sudden.
- Radio range: the tested radio range proved to be not evenly distributed. With the same distance and different orientation, reading differed.
- Overall inconsistent behaviour: sometimes the smart sign did not receive anything over the radio, did not show anything or just stopped working. This occurred at random times and sometimes a reset helped and sometimes it did not help. There could be multiple causes, like battery depletion or no good initialisation of the system.

These weak points would surface more, when the main factor for failure would not be present. The main factor for the failure of the demo was the lack of the appropriate amount of smart signs for this area. This resulted into a sparsely populated network, where signs did not have enough range to communicate properly with each other. Although the range was tested before, it turned out to be insufficient. The most likely causes of this difference are mentioned below.

- The orientation of the smart signs was not taken into account, while constructing the smart signs for the demonstration.
- The orientation was not tested before and could have led to smart signs being out of range.
- The range tests done on beforehand also did not include the housing. This could also jam the smart sign signal and cause signal loss.
- Finally, the positioning was not completely the same as the test set-up, which also could have affected the signal coverage.

Summarising, the demonstration went wrong, due to unforeseen behaviour, that could not be tested, because of time pressure created by the problems stated above.

# 10 Discusison

It is already stated in chapter 9 that the demonstration did not turn out to be a success. One. However, there are notes to this failure and other, not so visible at first sight, results are positive.

## 10.1 SVG conference demonstration

The demonstration goals, determined on beforehand, were not met afterwards. However, a personal demo for a mentor did work to some extent. It showed that the parts that operate on personal computers did work. The only weak link in the chain turned out to be the smart signs network. While reading the previous chapter one can see all the problems, but the positive side is that it almost worked and that next time a demo is given, the system can and will be tested more thoroughly in advance and will work almost for sure.

## 10.2 Other results

Apart from the demonstration at the conference other results were made. Especially on the personal area the results are positive. While testing the system during the implementation, the system component proofed to be working. The system also proved to be working concerning a small network of test smart signs (initial smart signs, see chapter 5). This shows that the system really works and that it has potential to work in bigger environments.

There is also a lot learned from the process of developing for embedded devices. This learning process ranges from taking into account the limitations of the device while designing, to actually implementing the system. While working on this subject also more insights were gained concerning working under time pressure and working at a company. This project also shows that, although it is good to have tight schedules, one has to stay realistic about a project schedule. If a schedule is too tight, it is likely to have a bad influence on the project, even if the people in the project work their hardest.

Finally the chance to apply gained knowledge also was a success. The knowledge about embedded devices helped in the process of making decisions as well as in the implementation process. Also the programming knowledge for personal computers was put to practice. Other knowledge that was put to practice was the process of going from a vague high level description, to a low-level implementation.

# 11 Conclusions and Recommendations

## 11.1 Conclusions

The concept of ambient intelligence is materialising and is going to materialise even more in the near future. Smart Signs are part of this vision. The concept of the Smart Signs is very viable to be materialised in full scale in the near future. In the present the materialisation in the form of a prototype proved to be a success. The designed system proved to be working on existing hardware, although there still are some hurdles, which have to be overtaken.

The designed and implemented Smart Signs system is a good basis for future, resource lean, environments, but is not yet capable enough to act as a solid platform to test the concept with users. Due to time constraints, the system could not be engineered and will need some testing to become a solid platform. However, this engineering will not be a long process, because the Smart Signs system is close to being finished.

### 11.1.1 The best suited realisation

The research question was the following: What is the best-suited realisation of the Smart Signs system, given the provided platform? The answer to this research question is described in this report, but can be summarised to the following: the best-suited realisation of the Smart Signs system is a combined central and distributed approach. The aspiration is to do as much as possible in a distributed way. However, given the provided platform and to improve the manageability, it was necessary to put central aspects in the system. To summarise, the following choices were made concerning the different functionalities of the Smart Signs system:

- *Guiding:* Predictive store, where a guide path is flooded with guide information to have a fast response time and to keep the stored data dynamic and within the bounds of a resource lean environment.
- *Printing:* Zone printing, where a virtual zone is drawn around the printer. When a user enters this zone the documents, chosen to be printed at an earlier time, are printed before the user reaches the printer.
- *Showing information:* Partial store, where information all information is stored on a central server and a cache of that information is stored at a Smart Signs for a period of time. This cache is filled when a user arrives at a sign, or through strategic flooding.

### 11.1.2 Objectives

Apart from the research question, other objectives were also stated and grouped in personal, Océ and Smart Surroundings objectives. All the personal objectives were met. A lot is learned while designing and implementing the Smart Signs system, concerning embedded devices, working in a company and applying gained knowledge. The objectives, which Océ and the Smart Surroundings group had envisioned, were not completely met. Although knowledge is gained, through this project, about Smart Signs, the demo was not made (see the practice chapter). This was due to the hardware platform and the tight schedule. In the near future, this goal can still be met in another demonstration.

## 11.2 Recommendations

The main recommendation is to further test and engineer the developed Smart Signs system, until it becomes a solid platform. This step should not take long, because the system is almost ready as a prototype. When this is done, user evaluation tests can be done. There are also some points, which can be improved concerning the Smart Signs system. Some of these concern technical aspects and user interfacing aspects:

### 11.2.1 Technical

Some technical recommendations can be made concerning the Smart Signs system. To begin with, it is recommended using more capable devices in the future. These devices are currently limited in storage and processing power. When the system is going to be used in bigger environments, more performance is needed. This however should not be overdone, resulting in big and energy devouring devices. The devices also have to be able to blend into the environment. The technical improvements should therefore be within the boundaries of energy consumption. An example improvement could be to reduce the latency between two smart signs. It is desirable to have this faster, because right now this is the bottleneck of the overall communication network. Another improvement could be to enlarge storage, which enables the possibility to do more in a distributed way.

Other technical recommendations concern the possibilities of the Smart Signs system itself. In the future one can add a more precise positioning platform. This requires an improvement in the platform as well as in the Smart Signs system. If a more precise position is known, the system can react more reliably on user behavior. This leads to the recommendation to use behavior recognition. This will improve the reaction time and will make the behavior of the Smart Signs system more intelligent.

### 11.2.2 User Interface

The communication to the user can also be improved. The recommendations for this area begin with realistic and easy added functionalities and ends with more futuristic recommendations. First guiding is described, then the tag is discussed and finally the smart signs.

Currently a request for guidance can be done by connecting to a socket. To begin with, a good web site should be made to improve the user interface. Placing a guide request can be made even more simple, by letting the Smart Signs system initiate the guidance process, by detecting a new user or a certain kind of behavior that can be linked to the need of a guidance.

Concerning a tag also some recommendation can be made. Now the tag is a passive element with no interface. This could be changed in the future. A button can be added to turn the tag on and off, or put it in a different mode, where the user is tracked or not. This should soften the privacy downsides of being tracked by the system. The tag can also be equipped with a screen and more interaction and feedback functions. In this way more interaction with the tag is possible, enabling all kinds of possibilities. For example, the tag can then be an interface for requests for guidance and more detailed and personal information. The question is however whether this extension is going to be an improvement. I believe that the environment (and not the tag) should contain as much functionality as possible. In that way the user does not need to adapt to the environment, but vise versa. People also do not want to keep big devices with them all the time.

Finally there are some recommendations for the smart signs itself, concerning the user interface. To begin with, the screen should be bigger and clearer. Then the users do not have to put so much effort into reading directions and information. More futuristic applications of the smart signs user interface could be using the smart signs as an interface for requesting data and guidance. The interface could also be extended to other than visual information provision. An example of this is the usage of an audio queue for guidance, like in car navigation.

# Reference list

| | |
|---|---|
| **[AMAN]** | T. J. Hofmeijer; “AmbientRT 0.5 Manual”; Enschede, Netherlands; 2005 |
| **[AMOD]** | http://www.agilemodeling.com |
| **[ART]** | T. J. Hofmeijer, S. Dulman, P. Jansen, P. Havinga; “Ambientrt - real time system software support for data centric sensor networks”; Melbourne, Australia; 2004 |
| **[CRIM]** | http://www.crimsoneditor.com/ |
| **[DIJK]** | http://ciips.ee.uwa.edu.au/~morris/Year2/PLDS210/dijkstra.html |
| **[EMIC]** | http://www.emmicroelectronic.com/products.asp?IdProduct=186 |
| **[HSSN]** | NOST network, “How Smart is Smart in the Netherlands?”; location unknown; 2005 |
| **[LMAC]** | L.F.W. van Hoesel, P.J.M. Havinga; “A lightweight Medium Access Protocol (LMAC) for Wireless Sensor Networks”; Enschede, Netherlands; date unknown |
| **[MGC]** | http://mspgcc.sourceforge.net/ |
| **[MGW]** | http://www.mingw.org/ |
| **[MSP4]** | http://focus.ti.com/docs/prod/folders/print/msp430f149.html |
| **[MSP6]** | http://focus.ti.com/docs/prod/folders/print/msp430f169.html |
| **[NOR]** | http://www.nvlsi.no/index.cfm?obj=product&act=display&pro=83 |
| **[NSN]** | L.F.W. van Hoesel, S. O. Dulman, P.J.M Havinga, H.J.Kip ; “Design of a low-power testbed forWireless Sensor Networks and verification”; Enschede, Netherlands; date unknown |
| **[OCE]** | http://www.oce.com |
| **[RFM]** | www.rfm.com/products/data/tr1001.pdf |
| **[SCW]** | A. Denver; “Serial Communications in Win32”; location unknown;1995 |
| **[SSUR]** | http://www.smart-surroundings.nl/ |
| **[SVG]** | http://www.svgopen.org/2005/ |
| **[WP5S]** | H. Hermes, R. Huis in’t Veld, A. Matysiak, A. Invanovic, F. Grobbe, A. Krohn, M. Lijding, J. Meijer, K.Sikkel; “Settings: the future is not what it has been….”; location unknown; 2005 |

# Glossary

## Definitions

| | |
|---|---|
| Ubiquitous office | An office, where computation is integrated into the environment. This results into having a intelligent office, which responds to users. |
| Ambient Intelligence | A vision of the future where people are surrounded by electronic environments, sensitive and responsive to people. |
| Smart Sign system | The prototype under development as a whole. This includes the Smart Signs network, gateway computer and the central server. |
| Static system | The prototype (as a whole), which does not respond to users. Here no sense of location is present and can only provide general information. |
| Semi-dynamic system | The prototype (as a whole), which responds to user locations. It can, for example, show personalised information. However, this prototype only offers one way information communication from the environment to the user. |
| Dynamic system | The prototype (as a whole), which acts almost the same as the semi-dynamic system. Here an added functionality is that the communication with the environment is both ways. Also person to person communication is available in this system. |
| Smart Sign | A wireless embedded device, which is used as a node in a wireless network. This device is used in the developed system to give the user information and to determine the location of a user. |
| Display node | The same device as a Smart Sign, but used only to provide users with information. |
| Tag | A wireless embedded device, which is a minimal version of a Smart Sign. This device can only act as a beacon by broadcasting a signal to all Smart Signs in the proximity. This device is carried by a user and is used to identify and locate a user. |

# Appendix

The appendix is enclosed after this report and contains the following subjects:

A. Initial planning
B. Actual execution
C. Requirements static system
D. Requirements dynamic system
E. Detailed explanation initial design
F. Detailed explanation changes Final architecture
G. Sequence Diagrams
H. SQL tables
I. CD content