

# User Interface Development and Data Visualization for Building Monitoring Systems

DIPLOMARBEIT

zur Erlangung des akademischen Grades

**Master of Science**

im Rahmen des Studiums

**Building Science and Technology**

eingereicht von

**Philip Ehrenfellner, B.Sc.**

Matrikelnummer 0551777

an der Fakultät für Architektur und Raumplanung  
der Technischen Universität Wien  
259.3 Institut für Bauphysik und Bauökologie

Betreuung: o.Univ.-Prof. Dipl.-Ing. Dr. Ardeshir Mahdavi

Wien, 05/2012

---

(Unterschrift Verfasserin)

---

(Unterschrift Betreuung)

# User Interface Development and Data Visualization for Building Monitoring Systems

MASTER'S THESIS

submitted in partial fulfillment of the requirements for the degree of

**Master of Science**

in

**Building Science and Technology**

by

**Philip Ehrenfellner, B.Sc.**

Registration Number 0551777

to the Faculty of Architecture and Urban Planning  
at the Vienna University of Technology  
259.3 Department of Building Physics and Building Ecology

Advisor: o.Univ.-Prof. Dipl.-Ing. Dr. Ardeshir Mahdavi

Vienna, 05/2012

---

(Signature of Author)

---

(Signature of Advisor)

# Erklärung zur Verfassung der Arbeit

Philip Ehrenfellner, B.Sc.  
Operngasse 32/12, A - 1040 Vienna

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit - einschließlich Tabellen, Karten und Abbildungen -, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

---

(Ort, Datum)

---

(Unterschrift Verfasserin)

# Acknowledgements

First I would like to thank my advisor Prof. Dipl.-Ing. Dr. Ardeshir Mahdavi whose support and helpful suggestions made this thesis possible. Furthermore I would like to thank Dipl.Ing. Dr.techn Szu-Cheng Chien who involved me in his research and brought me closer to the topic of UI development.

Above all I want to express my gratitude to my beloved family, especially my parents who made it possible to study and all my friends for supporting me during my studies and especially during writing this thesis. Special thanks to Clemens for his support to bring the source code to a higher level and speed.

# Abstract

The number of building monitoring and control systems is increasing steadily. In this thesis the importance of the user interface and the data visualization of such systems are discussed. The complexity of interface development and the desired user group's needs have been analysed in focus groups and a survey.

User interface development processes and different methods of digital and interactive data visualizations are discussed in terms of the web based building monitoring system, which is developed at the Department of "Building Physics and Building Ecology" of the University of Technology Vienna in context of the research project "Ubiquitous dynamic building performance monitoring".

A number of use-cases on behalf of facility managers are discussed and solved with an intuitive web interface. The modular system is based on the "Google Web Toolkit" (GWT) framework and allows a high flexibility for developers and users. The interactive data visualization was developed in the authoring framework "Unity" and shows a new way of handling data and accessing it easily.

A 3D model of the Department of Building Physics and Building Ecology of the Vienna University of Technology is combined with real time indoor data visualisation of temperature, windows status, and occupancy. The user can virtually walk around and access the data of each room in an intuitive way.

In addition, the 3D model is a preparation for upcoming augmented reality systems like "Google Glasses" presented in April 2012.

# Kurzfassung

Die Anzahl und Bedeutung von Gebäudemonitoring und -steuerungssystemen nimmt ständig zu. In dieser Arbeit werden die Benutzeroberfläche (User Interface) und die Datenvisualisierung von solchen Systemen diskutiert. Die Komplexität von "Interface"-Entwicklung und die Anforderungen der Nutzergruppen wurden anhand von Fokusgruppen und einer Umfrage analysiert.

Aktuelle Prozesse der Entwicklung von Benutzeroberflächen und unterschiedliche Methoden zur digitalen und interaktiven Datenvisualisierung werden anhand des webbasierten Gebäudemonitoringsystems, welches am Institut für Bauphysik und Bauökologie an der Technischen Universität Wien im Zuge des Forschungsprojektes "Ubiquitous dynamic building performance monitoring" entwickelt wird, analysiert.

Eine Anzahl von Fallbeispielen von Gebäudemängern werden diskutiert und mit der intuitiven webbasierenden Benutzeroberfläche gelöst. Das modulare System basiert auf "Google Web Toolkit" (GWT) und erlaubt eine große Flexibilität für Benutzer und Entwickler. Die interaktive Datenvisualisierung wurde in der Entwicklerumgebung "Unity" umgesetzt. Es zeigt einen neuen Weg Daten einfach für Gebäudebenutzer und -manager zugänglich zu machen.

Ein 3D Modell des Institutes für Bauphysik und Bauökologie an der Technischen Universität Wien ist mit Echtzeitinnenraumdaten verbunden, wie zum Beispiel Temperatur, Fensterstatus oder ob ein Raum besetzt bzw. belegt ist. Der Benutzer hat die Möglichkeit in dem Modell virtuell herumgehen und gleichzeitig die Daten eines jeden Raumes anzeigen zu können.

Weiters ist das Modell eine Vorbereitung für ein zukünftiges Augmented-Reality-System wie zum Beispiel die im April 2012 vorgestellte "Google Brille".

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Overview . . . . .	1
1.2	Motivation . . . . .	2
1.3	Background . . . . .	2
1.4	Problem Statement . . . . .	3
1.5	Structure of the Master's Thesis . . . . .	4
<b>2</b>	<b>User Interfaces</b>	<b>5</b>
2.1	Definition . . . . .	5
2.2	UI Types . . . . .	5
	Command Line Interface . . . . .	6
	Graphical User Interface . . . . .	7
	Web User Interface . . . . .	9
	Touch User Interface . . . . .	9
2.3	Usability and Design . . . . .	11
	Signals . . . . .	11
	Human Factors of Interaction . . . . .	12
	Layout Structure . . . . .	12
2.4	Development . . . . .	13
<b>3</b>	<b>Data Visualization</b>	<b>15</b>
3.1	Definition . . . . .	15
3.2	Visual Display . . . . .	15
	Visualizing Patterns . . . . .	17
3.3	New Ways, New Technologies . . . . .	17
<b>4</b>	<b>Methodology</b>	<b>20</b>
4.1	Focus Groups . . . . .	20
4.2	Use Cases . . . . .	25
	Situation . . . . .	25
	Cases . . . . .	25

4.3	User Interface . . . . .	29
	Modules and Widgets . . . . .	31
	Desktop . . . . .	31
	Graphs . . . . .	32
	Visualization . . . . .	34
	Export . . . . .	34
	Log Files . . . . .	34
	Accounts . . . . .	34
4.4	Technology . . . . .	36
	Google Web Toolkit . . . . .	36
	Unity . . . . .	37
<b>5</b>	<b>Prototype Evaluation</b>	<b>39</b>
5.1	Objectives . . . . .	39
5.2	User Interface . . . . .	39
5.3	Data Visualization . . . . .	41
	Numerical Visualization . . . . .	41
	Status Change Visualization . . . . .	42
	Color Visualization . . . . .	42
5.4	Data Model . . . . .	42
<b>6</b>	<b>Conclusion</b>	<b>45</b>
6.1	Lessons Learned . . . . .	45
6.2	Future Research and Development . . . . .	46
<b>A</b>	<b>Floor Plans</b>	<b>47</b>
A.1	Department of Building Physics and Building Ecology . . . . .	47
A.2	Sensors . . . . .	49
<b>B</b>	<b>Source Code</b>	<b>50</b>
B.1	PHP Source . . . . .	50
B.2	Unity Java Scripts . . . . .	51
	Global Scripts . . . . .	51
	Visualization Scripts . . . . .	53
	<b>Bibliography</b>	<b>57</b>



# List of Tables

2.1	Interaction Methods and their Control Parameters . . . . .	6
4.1	User Survey Information: 134 participants (Chien et al., 2011) . . . . .	21
4.2	Three different interfaces for building monitoring and control (Chien et al., 2011) . . . . .	21
4.3	Use case examples (Zach et al., 2012) . . . . .	28
5.1	Visualized real time data in the 3D model . . . . .	40
A.1	List of sensors and their acronyms . . . . .	49

# List of Figures

1.1	Oil and Gas price from 1991 to 2008 (energiesparen-im haushalt.de, 2012) .	3
2.1	CLI   OSX Terminal . . . . .	7
2.2	GUI   OSX Lion . . . . .	8
2.3	WUI   Google Maps . . . . .	9
2.4	TUI   Apple iPhone, iOS . . . . .	10
2.5	Design as a Communication-model (Heufler, 2004) . . . . .	11
2.6	Explorative Design . . . . .	13
2.7	Flexible Approach to Concept Generation and Selection (Buxton, 2007) . .	14
3.1	Graphical Disarrangement   “Lie Factor = 0“ but misleading scaling . . . .	16
3.2	Carpet-contour plot showing lighting electricity consumption against hour of day and day of week (for the most the week in which the fault occurs) (Rafferty and Keane, 2011) . . . . .	17
3.3	Wikitude Augmented Reality Browser . . . . .	18
3.4	Google “Project Glass“ - Augmented Reality Glasses . . . . .	19
4.1	Participants’ level of interest in energy use information of different systems (Chien et al., 2011) . . . . .	22
4.2	Participants’ level of interest in indoor environment information (Chien et al., 2011) . . . . .	22
4.3	Participants’ level of interest in outdoor environment information (Chien et al., 2011) . . . . .	23
4.4	Participants’ level of interest in occupancy information (Chien et al., 2011) .	23
4.5	Participants’ level of interest in building systems information (Chien et al., 2011) . . . . .	23
4.6	Participants’ preference for hardware usage to access data of the system (Chien et al., 2011) . . . . .	24
4.7	Participants’ views on the problems associated with interface systems for building-related information (Chien et al., 2011) . . . . .	24
4.8	UI Workflow Structure . . . . .	30

4.9	Mockup Graph Module: Stack Panel, Main Window (incl. grid and tabs), Fixed Widget (basic search view) . . . . .	31
4.10	Search function: basic view . . . . .	32
4.11	Search function: advanced view . . . . .	32
4.12	Graph widget: basic / advanced view . . . . .	33
4.13	Favorite widget . . . . .	34
4.14	Mockup Log Files Module: search-bar at the top and table of logs . . . . .	35
4.15	Mockup Account Module: user administration, interface for editing and creating accounts . . . . .	35
4.16	GWT in Eclipse   GWT Designer . . . . .	36
4.17	Unity   Development Environment . . . . .	37
5.1	User Interface of the 3D data model with all parts “Off“ . . . . .	40
5.2	User Interface of the 3D data model with all parts “On“ . . . . .	41
5.3	Color-bar of the radiator temperature ranges . . . . .	42
5.4	Screen shot of the real time data model with all parts “Off“ . . . . .	43
5.5	Screen shot of the real time data model with all parts “On“ . . . . .	43
5.6	Data Connection Model: Database, Connector, Object Controller, Anima- tion Scripts . . . . .	44
A.1	Floor Plan “Department of Building Physics and Building Ecology“ . . . . .	47
A.2	BPI Sensors . . . . .	48

# CHAPTER 1

## Introduction

### 1.1 Overview

Great advances have been made in the area of building monitoring systems over the last years and can significantly affect the indoor conditions, the energy usage and the occupants' comfort. Based on new Information Technology (IT) Systems it is possible to calculate accurate models of the conditions inside the building and therefore more precisely control the indoor environment.

Nevertheless, the specific research about the requirements of user interaction and the user interface of such systems have not been formulated in a rigorous and reliable manner (Chien and Mahdavi, 2008).

This thesis discusses the user interface of the building monitoring system developed at the department of Building Physics and Building Ecology of the Vienna UT in context of the research project "Ubiquitous dynamic building performance monitoring". The second section of this work presents a 3D real-time data visualization of the above mentioned department, as a preparation for augmented reality.

The first part of this thesis deals with the theory of User Interface Development and Data Visualization. The different types of user interfaces, the usability and design, the design process, important factors of data visualization and upcoming trends are discussed. In a second step a set of the use cases which are based on a survey and focus groups, an analyses of the user interface and the results of the 3D real time data visualization are presented.

## 1.2 Motivation

The impending digitalisation particularly in the building monitoring and control sector and the introduction of more and more sensors especially in sophisticated buildings make it imperative to develop new and simple user interfaces for the occupants. The management and analysis of this data poses as an increasingly difficult challenge in their day to day work. Only a well thoroughly planned and designed system can avoid frustration of the users and sensibilise them to the new systems (Christie, 1985).

The goal of this research is to show the importance of the user interface and the connected data visualization. The understanding of the designated users in terms of building control and monitoring systems is indispensable to solve this problem. This work should enhance the knowledge related to this field and highlight the importance of further research and development.

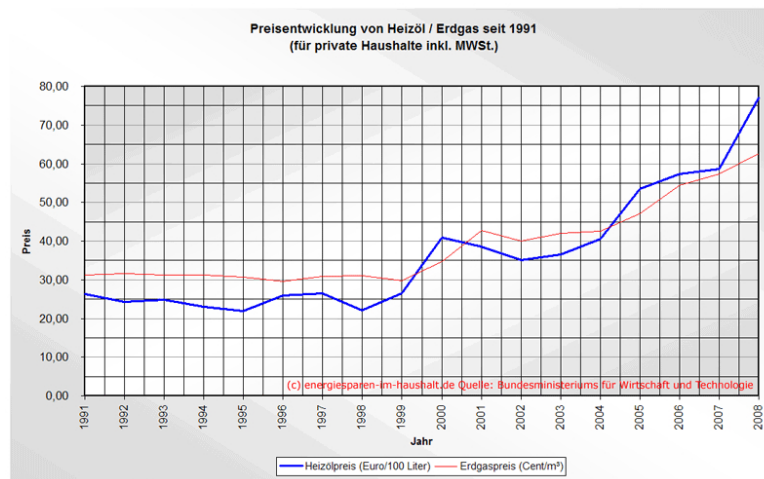
## 1.3 Background

The increasing importance of building control and monitoring systems is based on several reasons. The potential of compensating the rising oil- and energy costs (see Fig. 1.1) is one of the most frequent discussed ones. The European Commission promotes low-energy buildings and renewal energies (Commission, 2009) to improve the green-footprint in the member countries. More than 12.000 of 20.000 low energy houses have been built in Austria and Germany alone, but the rate in other countries is rather low.

The importance of this topic is underlined by a study of the International Energy Agency (IEA) (Lausten, 2008), which states that buildings are responsible for approximately 40% of the energy consumption, and about 36% of the CO<sub>2</sub> emissions in the European Union.

Another important factor is the control and maintenance of a modern building. The energy management of large buildings such as office towers or universities is influenced by numerous factors and cannot be improved without such a system. A monitoring network provides a wide set of benefits (Zach and Mahdavi, 2010):

- Energy optimization through improved management of technical building systems.
- Increased awareness of building users regarding their impact on buildings' energy use.
- Early detection (and treatment) of deficiencies and malfunctions in energy systems and devices, thus effectively supporting a preventive maintenance regime.



**Figure 1.1:** Oil and Gas price from 1991 to 2008 (energiesparen-im haushalt.de, 2012)

- Successive building performance improvement and optimization via the analyses of dynamically updated building energy and performance data bases.
- Long-term accumulation of empirical information on buildings' energy and environmental performance toward improving the design, construction, and operation of existing and new buildings.

The developed building monitoring infrastructure provides a flexible base to validate simulation models and to realize simulation-based control strategies. Finally, the model helps to create a simulation-powered based building system control strategy (Zach and Mahdavi, 2010).

At the moment two different buildings, the in 2010 completed “Lehrtrakt“ and the more than 100 years old main building, are partly equipped with monitoring infrastructure. This work will focus on the environment of the department of Building Physics and Building Ecology situated in the old building in terms of 3D real time data visualization.

## 1.4 Problem Statement

An easily understandable and usable building monitoring and control interface has not emerged so far. The systems available today are limited in functionality, effectiveness and usability (Karjalainen and Lappalainen, 2011). The increasing distribution of smart-phones and the permanent connection with the web over the the course of the last decade opened up a complete new way of communication and the connection of the occupants and their environment. The user interfaces of software have improved incrementally over the last decade in order to be easily understandable for everyone. Even though the

computer and smart phones are omnipresent in our daily life. Nevertheless, the sector of building software has not caught up on this trend.

The Vienna University of Technology consists of several buildings and has different facility managers who are responsible for their maintenance. The system needs to be available any time and anywhere to enable a smooth work flow. Though there are several people who will use it frequently, the user interface and the data visualization have to provide a certain level of personalisation for each user.

## **1.5 Structure of the Master's Thesis**

This work is structured into two main research fields, user interface development and data visualization. The theoretical part deals with the different user interface types and their development processes. Data visualization and its new technological possibilities supplement this part.

A discussion of the user interface, the actual system, which has been developed as well as suggestions for additional features are presented in the first part of the practical section. The prototype of a 3D real time data visualization, developed in Unity is presented as a preparation for augmented reality applications.

# CHAPTER 2

## User Interfaces

### 2.1 Definition

*“The User Interface is the part of a computer and its software that people can see, hear, touch, talk to, or otherwise understand or direct. The User Interface (UI) has essentially two components: input and output.” (Galitz, 2007)* The Interaction with the Computer occurs in multiple ways and involves different senses and communication skills of the user like taking/listening or reading/typing.

This work focuses on user interfaces in terms of Information Technologies, mainly web development, and will not discuss other forms of human-machine interaction.

### 2.2 UI Types

The technological enhancement and new requirements have resulted in different ways of human-computer interaction (HCI). Table 2.1 shows the possible “interaction methods” and their “control parameters”. A single method that could provide all needed interaction without combining with another one has not occurred so far. Just the combination provides a high usability for a major group of users.

The hardware interfaces like mouse or keyboard provide a high flexibility in terms of functions but also require a high knowledge by the interactor. They do not support multiple point interaction and present the largest distance between the digital and the analogue world. The further development of the mouse to the touch pad allows simple gestures of zoom or rotation.

At the moment the very famous touch interfaces are closing this gap and allow a very



**Table 2.1:** Interaction Methods and their Control Parameters

Interface	Supported UI Interaction
Hardware	
-mouse	-single point interaction
-keyboard	-text
-touch pad	-simple global gestures
Touch / Multi-touch	
	-multiple point interaction
	-direct interaction: nearly no barrier between digital content and the user. <i>“Touch the content”</i>
	-text via “digital keyboard” on the screen
Natural Interaction	
-gestures (visual / haptic)	-multiple interaction points via tracking hands, fingers or gestures
-eye tracking	-analogue commands via tracking the viewing direction
Speech recognition	
	-text
	-“spoken commands”

close contact to the digital content. Nevertheless, the user has to learn the different parameters.

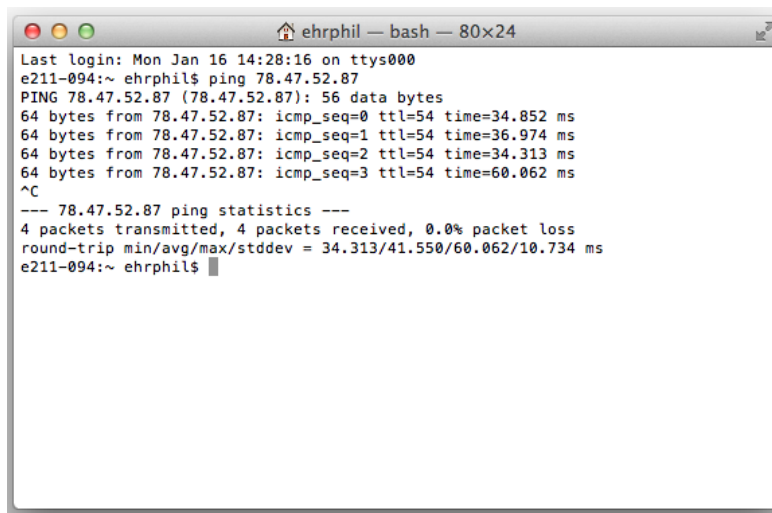
The upcoming technologies of “natural interaction” (gestures) and “speak recognition” can make the learning of the interaction obsolete, but they are still at the beginning of their development. In future the systems will take over the a big part of the data and interaction processing (for example: typing in content or browsing for a file) that is actually done by the user.

In the following part the four main types are discussed. They are sorted by their appearance starting with the oldest to the newest one.

## Command Line Interface

The Command Line Interface (CLI) is a “text-only interface” and performs task by typing in commands (see Fig. 2.1). Nowadays this system is used if there are not enough resources for a Graphical User Interface (GUI) (see “Graphical User Interface“, page 7) or if a high number of queries or commands can be entered more rapidly as text than with a pure GUI.

The CLI is mostly used by programmers, (system) administrators as well in the scientific area (comp. Wikipedia, 2012e). For non advanced users this system represents a major barrier since it requires knowledge about the different commands and their syntax.

A screenshot of an OSX Terminal window. The title bar shows a home icon, the name 'ehrphil', and the command 'bash' followed by the window size '80x24'. The terminal text shows a login message, a ping command being executed, four successful ping responses with varying times, and a summary of ping statistics. The prompt 'e211-094:~ ehrphil\$' is visible at the bottom.

```
Last login: Mon Jan 16 14:28:16 on ttys000
e211-094:~ ehrphil$ ping 78.47.52.87
PING 78.47.52.87 (78.47.52.87): 56 data bytes
64 bytes from 78.47.52.87: icmp_seq=0 ttl=54 time=34.852 ms
64 bytes from 78.47.52.87: icmp_seq=1 ttl=54 time=36.974 ms
64 bytes from 78.47.52.87: icmp_seq=2 ttl=54 time=34.313 ms
64 bytes from 78.47.52.87: icmp_seq=3 ttl=54 time=60.062 ms
^C
--- 78.47.52.87 ping statistics ---
4 packets transmitted, 4 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 34.313/41.550/60.062/10.734 ms
e211-094:~ ehrphil$
```

**Figure 2.1:** CLI | OSX Terminal

## Graphical User Interface

The first commercial Graphical User Interface (GUI) for operating systems was developed by XEROX in the 1970ies. Apple was inspired by the concept and introduced “Apple Lisa“ with a GUI in 1983. Microsoft launched its “Windows 1.03“ two years later.

The breakthrough for the new human-machine-interaction took nearly ten years. This can be explained on the one hand with the missing computing power for an adequate implementation and and the fact that most software at that time was not built to support a GUI. The 16 bit computers started the today’s interface standard and with “Windows 3.11“ Microsoft became one of the biggest and most powerful operating system providers.

The interface-hardware, the first “pc-mouse“ was invented by Douglas C. Engelbart und William English at the Augmentation Research Center (ARC) at the Stanford Research Institute (SRI) in 1968. Not much attention was paid to the invention at that time because of the missing GUI. In the 1970ies XEROX further developed Engelbart’s prototype and released the first ball-mouse in 1972 (comp. Wikipedia, 2012f).

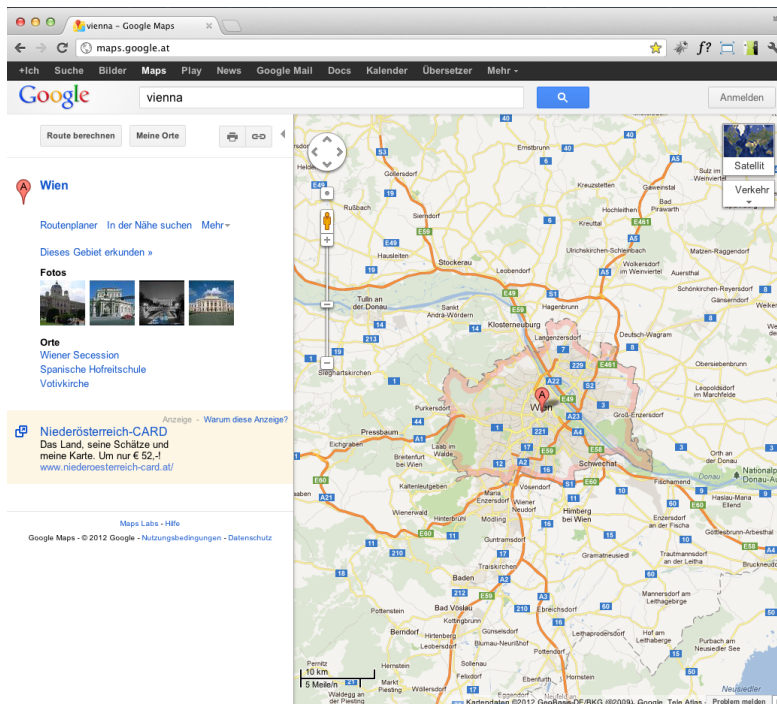
The standard “DIN EN ISO 9241-10 ff.“ (DIN, 2006) describes the requirements for a UI and is called “Ergonomics of Human System Interaction“. In part 10 “Dialogue Principles“ are formulated but do not refer to any situation, application, technology or use. This can be explained with the huge diversity of this topic and the specific requirements and different desired tasks of each program that does not allow a global definition.



**Figure 2.2:** GUI | OS X Lion

Seven mature points are listed:

- **Task Suitability**  
The system helps the user to fulfill the designated task.
- **Self-Descriptiveness**  
The user always has to know in which dialogue he is currently acting, and what interaction can be performed in which way.
- **Expectation Conformity**  
The context has to expect the users needs with regards to vocabulary, suitable feedback or structure of information (e.g.: menu).
- **Promotion of Learning**  
The dialogue has to be understandable for the user and is required to help as well as information has to be provided by the system.
- **Controllability**  
The user has to be able to start and control the dialogue in terms of direction and speed to reach the aim.
- **Error Tolerance**  
The dialogue should help the user to reach the aim despite of faulty entries by minimal correction of the user or the system itself.
- **Individualization**  
The user should be able to change the layout and design for personal needs.



**Figure 2.3: WUI | Google Maps**

## Web User Interface

The Web User Interface (WUI) is a subclass of a GUI. In the last decade this area got more and more important. Web-companies like Google and their web applications (e.g.: Google Maps (see Fig. 2.3)) changed the initial design structure that focused on the navigation and the presentation of information to a data processing UI like everybody is used to from desktop-programs.

It is not always clear where a web page ends and an application begins. Generally it can be said that a page is designed to provide information, an application to enable the user to interact.

## Touch User Interface

The Touch User Interface (TUI) is a mutation of a GUI that is designed to interact by touching with the user. The controller-devices, these are normally the mouse and a keyboard, are replaced by a “touch-sensitive“ display that interprets the user’s finger-interaction. The first touch-screen device has been released by Hewlett-Packard in 1983 but the breakthrough started more than two decades later with Apple’s iPhone in 2007 (see Fig. 2.4). The success can be explained with intuitive, finger-optimized GUI, and



**Figure 2.4:** TUI | Apple iPhone, iOS

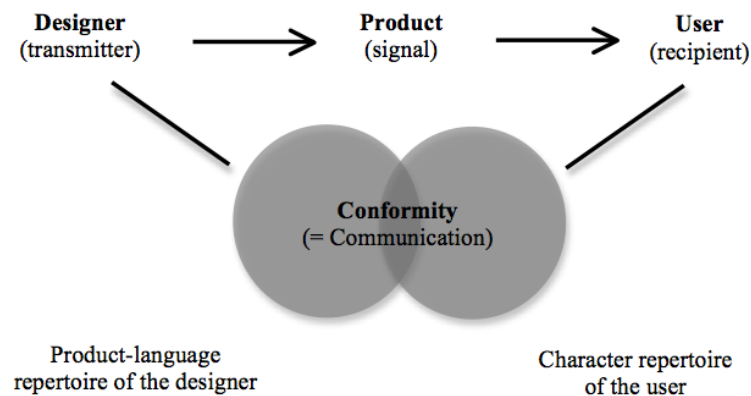
the multi-touch capable touch-screen.

The TUI has to be optimized for the average size of finger-tip and intuitive habits like scrolling by pushing upwards, to provide optimal usability (see “Usability and Design“, page 11). Studies show, that this new human-computer interaction is easily understandable and allows even elderly people who haven’t been growing up with a PC to understand the system more quickly.

Special designed “Touch Web User Interfaces“ are gaining more and more popularity. The rise of the smart phones and tablet PCs forced the web designer to adapt their designs for this type of hardware. New application programming interfaces (API)<sup>1</sup> between hard- and software and the technical progress of web-coding-language often make special designed applications (apps) already obsolete.

---

<sup>1</sup>*An application programming interface (API) is a specification intended to be used as an interface by software components to communicate with each other. An API may include specifications for routines, data structures, object classes, and variables (Wikipedia, 2012c).* The API can be language-depended (just available by using the same syntax of the programming language) or -independet (available via several programming languages).



**Figure 2.5:** Design as a Communication-model (Heufler, 2004)

## 2.3 Usability and Design

The usability of a product is on the one hand based on the optimal interplay of “Hardware“ and “Software“ and on the other one based on the quality of interaction with the user. To reach a high standard it is very important to understand the desired focus-group in several terms like age, field of work, desired basic knowledge about the product and others.

### Signals

*“Communication is however only possible if the sign repertoires of a transmitter (designer) and a receiver (user) are compatible. The designers’ task therefore has to be to translate a product’s functions into signals that receivers can also understand. This means that they have to study the target group closely and learn their “language“ first of all.” (Heufler, 2004)*

The “Communication Model“ of Meyer-Eppler (see Fig. 2.5) shows the three basic modules of communication: transmitter, signal and recipient. If there is no conformity in terms of the signals (e.g. icons) there is a high barrier for the users.

Every human being has learned a common knowledge about signals and habits during life. The color “red“ for example is always identified as to be aware of something. This does not mean that a designer is restricted to existing common signals or the users are not able to learn new ones. It just shows the importance of combining existing and new signals to keep the process of learning and understanding as simple as possible.

Especially for products which are not used frequently it is very important to remember

or at least get reminded quickly how to interact. This can be supported by highlighting important parts or possible actions, common-known icons, or the structure of the design (comp. Preece et al., 2002).

## **Human Factors of Interaction**

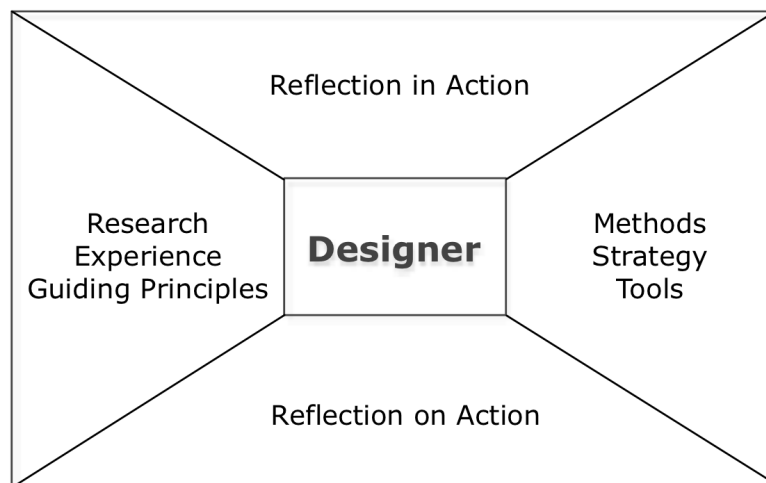
IT-Systems promise to increase efficiency, provide a basic and effective solution for every individual and improve the information processing. There is no adequate model to present a user-system interaction. For every product an individual model has to be designed in order to verify that the designer and the users are provided with the same information basis (Christie, 1985). A “common language“ can only be defined for a certain group of people.

Only qualitative research techniques can lead to the required knowledge. Especially in social behaviour quantitative research is not sufficient to understand the desired user group (Cooper et al., 2007). Focus groups of the desired user-group are a very good opportunity to understand the potential users. The direct discussion provides the product-developer with a deep insight and direct contact to their basic-knowledge, fears, habits and needs (comp. McDonagh-Philip and Bruseberg, 2009).

## **Layout Structure**

In web-development the structure and layout of a website has to adapt automatically to different screen resolutions. Therefore a few key facts have to be considered:

- Basic grid of content  
Therefore a simple grid, with the exact position of each content is designed. For a simple website this could be at the top the “Header“ containing menu and logo, in the middle the “Content-Area“ for all different types of information and at the bottom the “Footer“ for Legal Infos and copyright information.
- Menu structure (site map)  
The Menu is structured into main and sub-levels to simplify it for the user. The different links can be supported by icons.
- Screen resolution  
Websites can be designed in two different ways in terms of resolution:
  1. The website has a fixed resolution and will not change the layout or structure as soon as the size of browser window is changed.
  2. The website is designed for different screen resolutions and changes layout and/or structure based on the size of the browser window.



**Figure 2.6:** Explorative Design

Highlighting information or links are often realized by using different colours (see “Signals“, page 11) but a good thought through interface often makes this obsolete and simplifies the UI.

## 2.4 Development

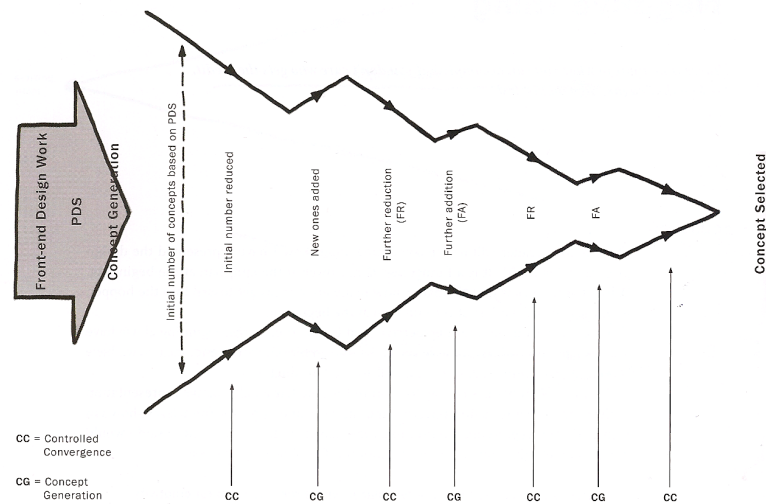
*“Every user is unique, too. What one person finds difficult, the next one won’t. The trick is to figure out what’s generally true about your users, which means learning about enough individual users to separate the quirks from the common behavior patterns.” (Tidwell, 2006)*

The evaluation of a problem, its definition and solution is not appropriate nowadays. In contrast to former standards, the development of a UI is an ongoing process of reflections and evaluations (see Fig. 2.6).

Nevertheless “Guiding Principles“ should be defined before the designing process itself takes off. They contain research on existing products and their evaluation on strength and weaknesses, existing standards and best practices to avoid mistakes of others. Overthinking the principles is always important but there is never a “perfect solution“. We live in exponential times and there is a very short life time of products and design. What seems perfect today, may be already outdated tomorrow.

Donald Schön (Edwards et al., 1996) points out the importance of “reflection-in-action“ in order for the designer to become a researcher in the field he or she is working in.





**Figure 2.7:** Flexible Approach to Concept Generation and Selection (Buxton, 2007)

During a design process this means to overthink the work and not to get stuck with standards and common design-principles. Combined thinking and doing is not bound to rationality and helps to develop a new design. When a solution is found, the “reflection-on-action“, as well as the critical-discussion afterwards points out new input and still existing weaknesses. This evaluation is normally performed in a group to collect several different opinions.

The design funnel (see Fig. 2.7) visualizes the above described design process. The initial number of concepts is not just narrowed down to the final one, there are occurring new ideas that are integrated in the ongoing design process. This approach is a very dynamic procedure to generate a concept.

## Data Visualization

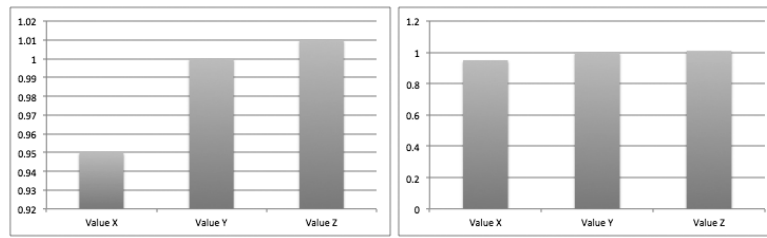
### 3.1 Definition

*“Data graphics visually display measured quantities by means of the combined use of points, lines, a coordinate system, numbers, symbols, words, shading, and color.” (Tufté, 2001)* Visualizing information is basically everything, as long as sufficiently organized (Friendly, 2009). Tables, graphs, point maps or even text or computer games are a kind of visualization. Subsequently just the graphical visualization of data will be discussed. In this context the main target is to communicate relations and changes of the given values clearly and easy to understand for the prospector.

### 3.2 Visual Display

The data sets are not all equal and there are nearly endless possibilities to graphically visualize data. *“The main goal [...] is to communicate information clearly and effectively through graphical means” (Friedman, 2008)*. It has to stimulate and get the attention of the viewer (Viegas and Wattenberg, 2011). Reaching these aims can only be achieved by knowing the target viewer group very well. Several different ways of visualizations like graphs are easily understood but may not always result in the best possible way. The designer has to make a decision what is the best for the desired focus group or gives the user a possibility to change the visualization to his preferences.

To stimulate and attract the viewer colors and graphics are still one of the best tools. For example a combination of population data and a map of the analysed area(s) help the user to understand and compare the values quickly and easily. Furthermore it is important to simplify the data set to a certain level and choose the right scale for the graphical



**Figure 3.1:** Graphical Disarrangement | “Lie Factor = 0” but misleading scaling

visualization.

The distortion in a data graphic is often used that a viewer misinterprets a given set of data. Edward Tufte (Tufte, 2001) defined two principles for graphical integrity:

*The representation of numbers, as physically measured on the surface of the graphic itself, should be directly proportional to the numerical quantities represented.*

*Clear, detailed, and thorough labelling should be used to defeat graphical distortion and ambiguity. Write out explanations of the data on the graphic itself. Label important events in the data.*

The accuracy of a graphic and the violation of the first principle is measured by the “Lie Factor” that is defined as followed:

$$LieFactor = \frac{sizeofeffectshowningraphic}{sizeofeffectindata}$$

If the factor equals one, the visualization is most likely representing the data set correctly. Beside there is still the possibility of the “graphical disarrangement” like changing the size of the labels or colouring and highlighting in a misleading way, that can lead the viewer to a wrong conclusion (see Fig. 3.1).

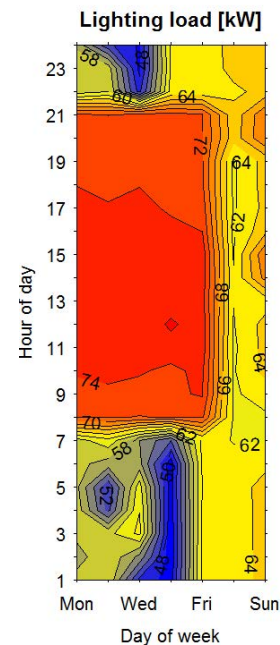
Summarized, the quality of data cannot be improved by adding ornamental hatching or false perspectives. The graphical presentation should be elegant and displayed and labelled in a correct scale (comp. Tufte, 2001).

## Visualizing Patterns

A well monitored building outputs a significant amount of data. *“Furthermore, there are interdependencies between multiple discrete (hour of day, day of week, etc.) and continuous (dry- bulb temperature, dew-point temperature, etc.) variables.”* (Raftery and Keane, 2011)

For visualizing these data sets bar-charts, histograms, line-charts and area charts are widely used. They are easy to generate and visualize but do not always fulfil the requirements. Users can misinterpret the data sets or do need a long time to understand them. The approach of three dimensional plots are a useful and easily understood possibility but they cannot be plotted in hardcopy. Beside it is often difficult to find out the exact position of a data point because of the perspective.

Combining these two visualization methods leads to the data visualization model of Raftery and Keane (Raftery and Keane, 2011). They combined the carpet plots with binning. The coloured plots have two axes and the overlaying contour lines help to identify each data point.



**Figure 3.2:** Carpet-contour plot showing lighting electricity consumption against hour of day and day of week (for the most the week in which the fault occurs) (Raftery and Keane, 2011)

## 3.3 New Ways, New Technologies

Smart phones, touch screens, GPS (Global Positioning System) and wireless internet access offer a new wide range of possibilities. Data can be connected with GPS-coordinates automatically to display location-based information.

In these terms, the term “augmented reality“ (AR) occurs frequently. It is a live, direct or indirect, view of a physical, real-world environment whose elements are augmented by computer-generated sensory input such as described above (Wikipedia, 2012d). The human computer interaction (HCI) can be dramatically enhanced with this new technology (Portalés et al., 2010).

The groundwork has already been established by research and technology but there is still a long way to go. At the moment this technology is only rarely used and is at an early stage of its development. This can be explained with the missing solution for



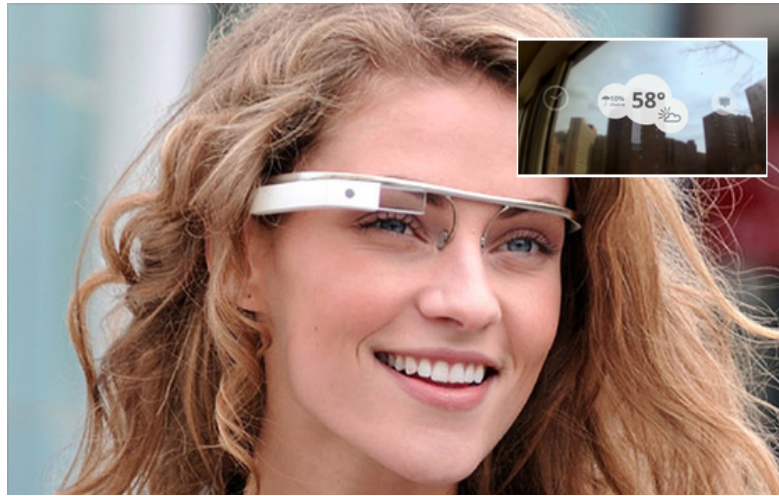
**Figure 3.3:** Wikitude Augmented Reality Browser

a proper interaction via natural gestures, eye tracking or speech recognition (see “UI Types“, page5).

“Wikitude“ (Wikitude, 2012), a start-up from Vienna developed a mobile application that aggregates all information of a user’s social media accounts and knowledge data bases (like Wikipedia) and visualizes them in context of the actual surrounding and location. (see Fig. 3.3) Smart phones are good and easy accessible devices for augmented reality systems, but there is still a big barrier between the “real“ and “digital“ world. The user has to interact with the device by touching with his fingers what leads to distractions of the user.

In 2012 Google presented its “Project Glass“, a new type of glasses including a head-up display (see Fig. 3.4) (Bilton, 2012a). The device includes the same technology as a normal smart phone, but instead of a touch screen, the head up display is used for visualization. A speech recognition and eye-tracking help to interact with the system (Bilton, 2012b). This early stage prototype shows what will can be expected in the near future.

The device allows to display actual weather information (see Fig. 3.4), navigation or dictate texts to send as Email or text messages (Google, 2012). The device has already been detected in reality but there are no public and reliable test how well the system is already working (derStandard, 2012). Nevertheless, this technology will dominate and may replace the smart phones by glasses like Google’s in near future.



**Figure 3.4:** Google “Project Glass“ - Augmented Reality Glasses

At the moment the tracking of a device via GPS or other technologies inside buildings is still a problem. Actual research with motion sensors closes this gap and will allow to navigate without a positioning system. The device is located on a certain point and from then on the gravity sensor in combination with the floor plan allows an accurate calculation of the position.

## Methodology

### 4.1 Focus Groups

In order to understand the desired user group and their expectations in terms of accessible values of a building monitoring system a user survey (questionnaire) with 134 participants (see Table 4.1) was conducted (Chien et al., 2011). The collected data includes energy use, external and indoor environment, occupancy and environmental systems.

- energy use: space heating, cooling, warm water, lightening, equipment
- indoor environment: room air temperature, relative humidity, air velocity, air change rate, CO<sub>2</sub> and VOC concentration (indicators of indoor air quality) and illuminance level
- external environment: weather conditions (sunny, cloudy, rainy), outdoor air temperature, relative humidity, wind speed
- occupant related information: presence, movement, number and location of occupants and their actions
- environmental control systems: HVAC (heating, ventilation, air- conditioning), windows, doors, blinds, and others

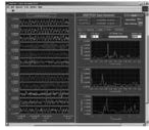


In addition focus groups, each with 24 participants, one session with experts and two with building occupants, were held to get a deeper understanding of the requirement context and the expectations of the desired users regarding building monitoring systems. (see “Human Factors of Interaction“, page 12)

Three different commercially available products for building monitoring and control

**Table 4.1:** User Survey Information: 134 participants (Chien et al., 2011)

Gender/Marital Status	Male: 43%; Female: 57%   Single: 68%; Married: 32%
Ages	16-20: 1%; 21-25: 22%; 26-30: 29%; 30-35: 26%; 36-40: 13%; 41-45: 3%; 46-50: 2%; 51-55: 2%; 56 and older: 2%
Residence	Austria: 50%; Taiwan: 50%
Education status	High school: 2%; College/University: 36%; Master: 53%; Doctor: 9%
Disciplinary Background	Design: 35%; Computer Science: 22%; Management: 10%; Art/Music: 8%; Social Science: 8%; Linguistics/Communications: 7%; other: 10%
User types	Occupants, guests: 72%; Experts (building operators, facility managers, system developers): 28%

**Table 4.2:** Three different interfaces for building monitoring and control (Chien et al., 2011)

Company Name	DIGITEXX digitexx.com	Agilewaves agilewaves.com	Oberlin oberlin.edu/dormenergy
Illustration			

systems were presented, discussed and rated by each group (see Table 4.2). The sessions lasted one and a half hours and were structured into five sequences (Chien et al., 2011):

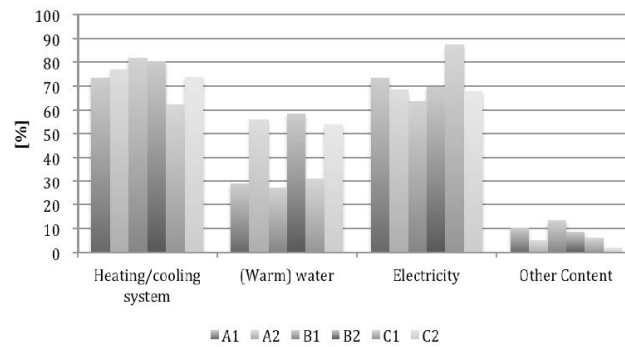
- i) completing a questionnaire
- ii) warm-up: general expression of participants' understanding of monitoring systems for buildings
- iii) Discussion: comments on the selected interfaces
- iv) personality profiling: game of personality mapping (McDonagh-Philip and Bruseberg, 2009) with selected interface products to elicit emotional responses to products
- v) brainstorming: developing interface functionalities and requirements

The gained data was analysed in terms of the above mentioned categories. The survey results are structured in the following manner:

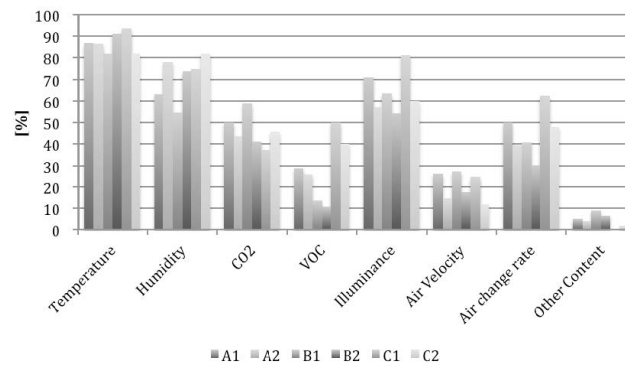


“A” denotes all users                      “1” specifies experts  
 “B” denotes users in Austria        “2” specifies building occupants  
 “C” denotes users in Taiwan

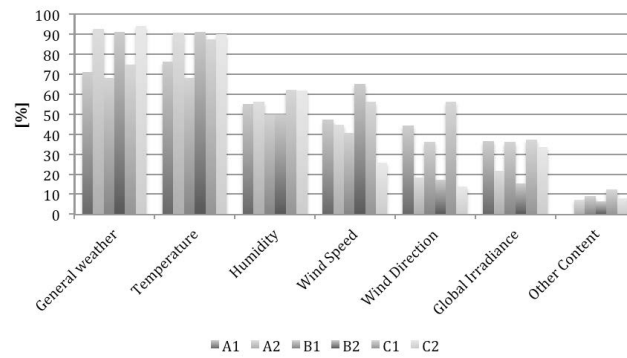
Thus, in combination six groups (A1, A2, B1, B2, C1, C2) are considered and the data streams are categorized into information streams in terms of energy use, indoor and outdoor environment, occupancy and environmental control systems. The results are shown in the Figures 4.1 to 4.7.



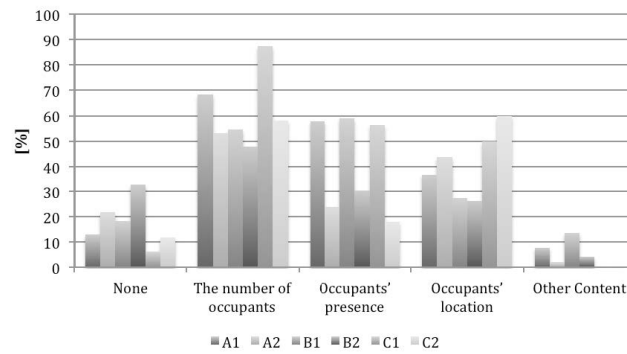
**Figure 4.1:** Participants’ level of interest in energy use information of different systems (Chien et al., 2011)



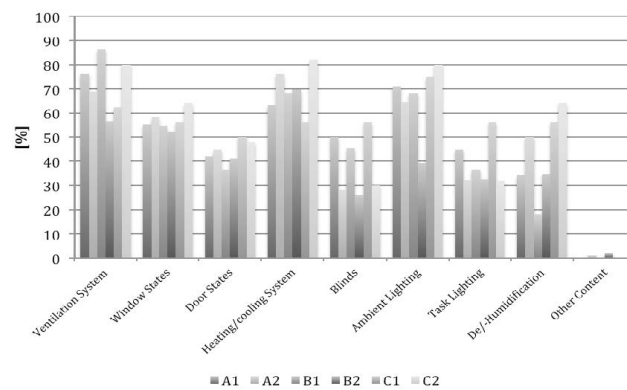
**Figure 4.2:** Participants’ level of interest in indoor environment information (Chien et al., 2011)



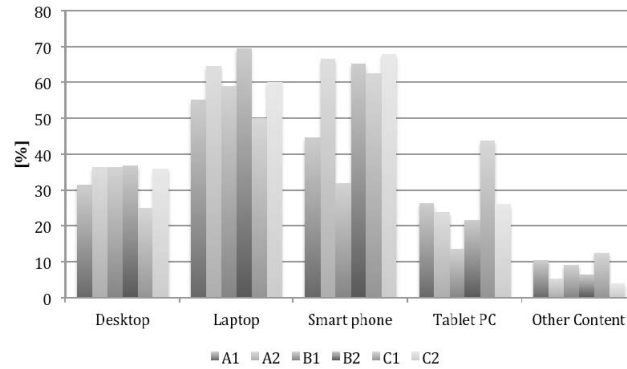
**Figure 4.3:** Participants' level of interest in outdoor environment information (Chien et al., 2011)



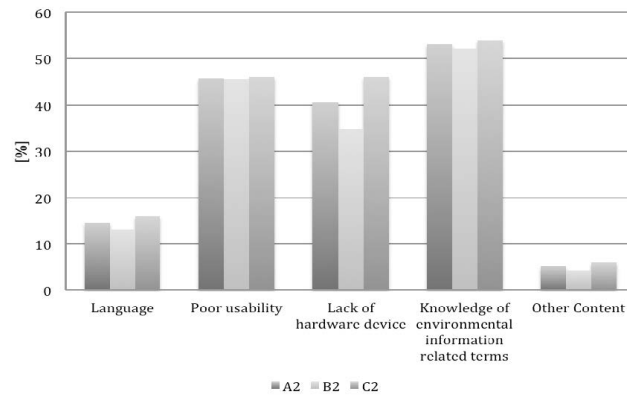
**Figure 4.4:** Participants' level of interest in occupancy information (Chien et al., 2011)



**Figure 4.5:** Participants' level of interest in building systems information (Chien et al., 2011)



**Figure 4.6:** Participants’ preference for hardware usage to access data of the system (Chien et al., 2011)



**Figure 4.7:** Participants’ views on the problems associated with interface systems for building-related information (Chien et al., 2011)

The survey results point out that building “experts” are much more interested in detailed technical information than occupants. For example in terms of outdoor conditions, they do consider detailed values like wind speed or global irradiance much more important than the general weather information (see Fig. 4.2). In comparison non experts are more interested in general information, for example indoor temperature (see Fig.4.2) or general weather (see Fig. 4.3). Also the differences of occupants information are significant between the two groups (see Fig. 4.4). Building experts tend to show more interest in detailed information like number of occupants as well as mentioned before.

The results show a high interest of both user groups in terms of energy usage (see Fig. 4.1). This raises up the question of the potential of user participation with regards to

energy efficiency and reduction. It has to be considered, that in this case control devices have to be implemented.

It can be assumed that a user interface for each group should have some significant differences, especially in data visualization. For occupants, the information may be visualized by icons, for example if it is raining, an icon shows rain. In comparison for building experts the wind speed has more weight than the general conditions. Furthermore the factor of mobility has a high relevance for both groups (see Fig. 4.6). The high interest in an UI for smart phones raises the question of providing two interfaces, one for “normal screens“ and one for “small“ phone screens (see “UI Types“, page 5).

## **4.2 Use Cases**

The focus groups (see “Focus Groups“, page 20) helped to structure use cases for the desired user-group of facility managers. Beside the conformity of signals (see “Usability and Design“, page 11) it was important to understand which values and functions of the system are more important than others. This knowledge is the key-information for the development of structure and hierarchy and further on for the UI design and data visualization.

### **Situation**

For the development of the use cases, the following situation was assumed:

All states of all devices in the building like if doors or windows are open/closed, if lights are on or off, if a room is occupied or not, etc. and the complete usage and status of the HVAC-System (Heating, Ventilation, Air-Conditioning) like the temperature of a room, a radiator or the humidity are known. As well we know the complete energy consumption of the building on the level of workplace. Overall, every important information in terms of energy-usage and user-behaviour is tracked.

### **Cases**

#### **1. Overview**

The overview-page is modular and customizable for each user. Individuality on the one hand helps to satisfy the users needs, and on the other to improve the individual work flow. The pieces of information for this layer are divided into three categories:

- **Overview of Outdoor Information**

Every focus group has rated the outdoor conditions (temperature, humidity, wind) as interesting values (see Fig. 4.3). The overview page gives the user a feeling about the actual values and problems. A quick comparison of the actual and past indoor conditions without knowledge about the outdoor conditions is impossible and leads to miss interpretation. The individual access to the outdoor values is important to address different user groups perfectly as already discussed above.

- **Overview of Indoor Information**

The indoor conditions in terms of temperature, lights, HVAC, CO<sub>2</sub>, VOC, doors, windows and occupation can be accessed in this layer (see Fig. 4.2 and Fig. 4.4).

- **Alerts and system information**

Actual important notifications pop up on a present area of the interface to be mentioned immediately by the user. Further notifications and errors are accessible via a separate layer (see 4. System Errors and Notifications)

## **2. Data Visualization**

The fast and simple access to the provided values and their visualization will be the most important feature of the system. The users will analyse and evaluate the data with these tools in most cases.

- **Graphs**

Dynamic, customizable graphs allow a high flexibility in terms of visualization and data ranges. Different value types can be combined and analysed. The user will be able to choose style, scaling and values of each graph according to his own preferences. Important is the fast renewal of the graphs in terms of changes to grant a good work flow.

- **3D Data Visualization**

A 3D real-time data visualization, in the optimal case, is accessible via augmented reality. A 3D visualization, if it is not realized in the way mentioned above, has to be web-based and implemented directly into the system. Installing additional components to access it has to be reduced to a minimum or if possible solved via standard web components like HTML5 or Java Script.

If web-based, this feature allows to walk through the desired monitoring area without leaving the office. Problems can be analysed more quickly and validation of them will be possible remotely.

In case of augmented reality the real time data input allows a very detailed and easy to handle analyses without a distraction by the IT device (e.g. Tablet or smart

phone). For example if there is an error notification, the system shows the exact way to the defect device via real time navigation.

### **3. Data Export**

The export function in comparison to the data visualization (Graphs) provides the function to conduct more advanced calculations and evaluations with programs that are not part of the system. The search function has to be intuitive and very flexible at the same time. All monitored values can be accessed, combined and exported to different file formats such as “XLS“, “CSV“ or “XML“.

### **4. System Errors and Notifications**

A Log-File tracks all system errors or notifications and can be browsed through. Important notifications are shown with a high presence in the UI. Different colors help to immediately identify the priority.

### **5. User Administration**

A user administration allows to create new accounts and manage the existing ones. The administration of the users allows role and group handling. Different groups are defined with different system and value accessibility.

### **6. View of Real-Time and Historical Data**

The view of different data (e.g. measured-values, zones, energy consumption, user-feedback) can be restricted or combined in several terms. The time is set to a certain past value (e.g. last hour) and will update automatically.

In Table 4.3 a few examples of use cases based on the above six points are formulated. The illustrated cases show common situations and problems a building monitoring system should cover.

**Table 4.3:** Use case examples (Zach et al., 2012)

Use case category	Use case examples
Visualize real time and historical data	<ul style="list-style-type: none"><li>- Show all temperature values in the zone X</li><li>- Show the electrical energy consumption in the zone X from 01/2012 to 04/2012</li><li>- Show all zones, which have a temperature above 24 degrees</li><li>- Show zones of dissatisfied users, the type and value of the reason (the reason is reported by the user)</li><li>- Show the top 10 energy consumer zones (energy by square meter) of non occupied zones</li></ul>
Show prediction of future energy needs/costs	<ul style="list-style-type: none"><li>- What are the energy needs/costs for the next few days (based on the weather forecast, etc.)</li><li>- On which days are which zones usually not occupied (e.g. by analyzing historical data. Most users are on holiday in calendar week x or on days x and y)</li></ul>
Give suggestions of what actions could optimize building performance	<ul style="list-style-type: none"><li>- Turn off the light and reduce the temperature in zones X</li><li>- Suggest window and door states (to reduce overheating and use natural cooling)</li><li>- Suggest window and door states for the best cross-ventilation with current or predicted outdoor weather conditions</li><li>- Suggest working times and days based on predicted work area conditions</li><li>- Suggest work place for mobile workers based on their requirements and the building performance</li></ul>

## 4.3 User Interface

The designated work flows (see Fig. 4.8) are designed with a flat hierarchy and a high intuitiveness. The user can access each function with a low number of actions (clicks) and is routed through a process by the system's design. The user interface is designed with an adaptive approach to extend its functions.

After the login on the start page, the interface is structured in 3 different parts:

- **Stack Panel Menu**

The stack panel on the left side is always active and displays the main menu points. The active one additionally shows its children to inform the user about the total amount of functions of this part. This design grants a low number of clicks and an easy understanding of the whole system. Although different screen resolutions do not impair its functions or usability. The users do know this type already well from Email clients like Microsoft Outlook or the mobile Application of Facebook. Especially in the last years the importance of this design was rising to handle the different resolutions of smart phones, tablets and PC monitors without major changes in the UI.

- **Interaction/Main Window**

The right side of the interface is used for the main window. This part uses about 85% of the screen width. It contains all interactions of the system, the widgets.

The window is structured by a grid that sizes the widgets automatically and attaches them to their position (see Fig. 4.9). This prevents overlapping of the active widgets and helps the user not to lose the overview. The grid will only be visible when a widget is moved. Based on the actual screen resolution the grid is adapted in terms of numbers of vertical and horizontal sub-divisions.

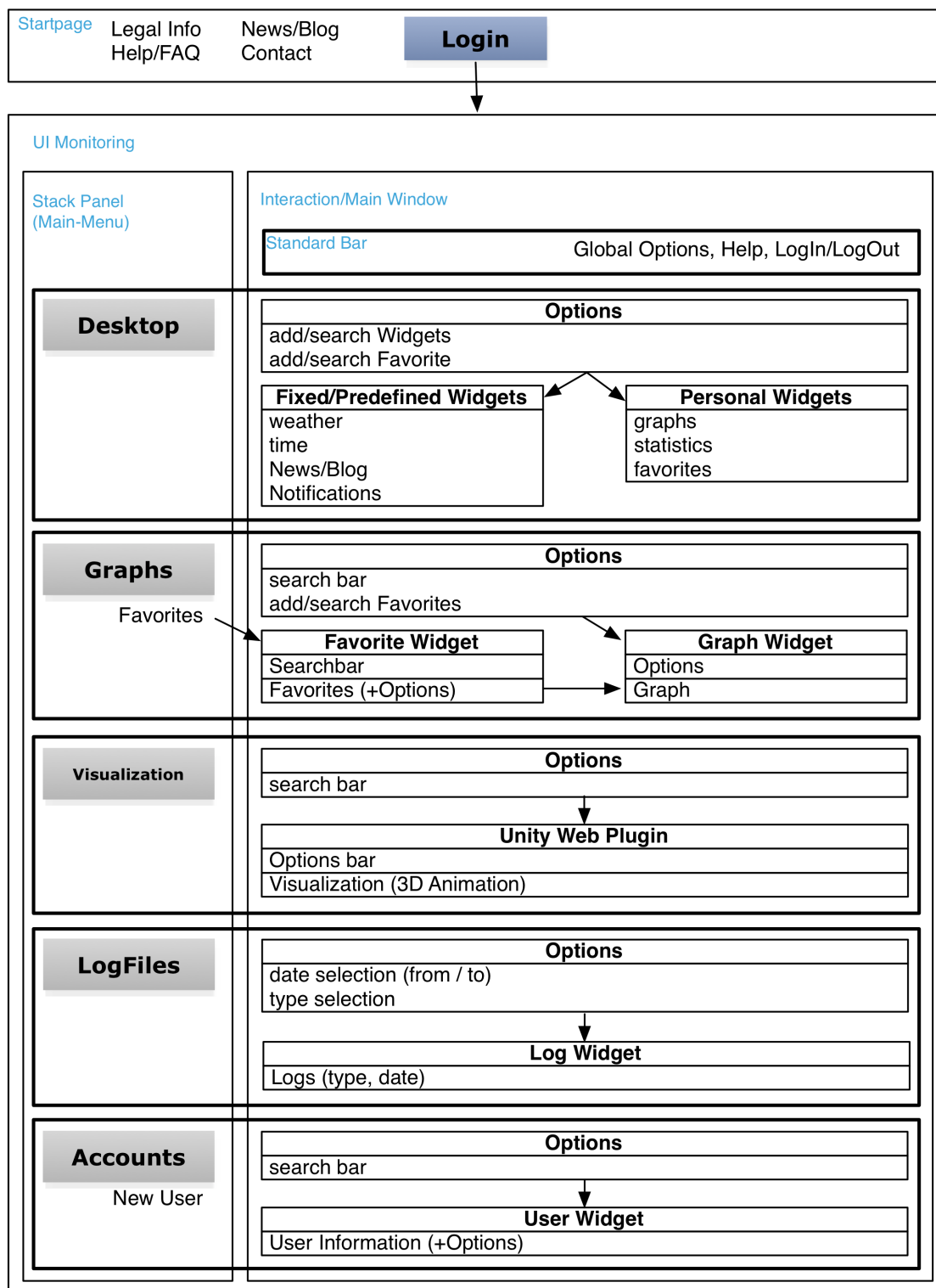
On the top a "Tab Bar" allows to create multiple screens to sort the widgets and extend the working area.

- **Widgets**

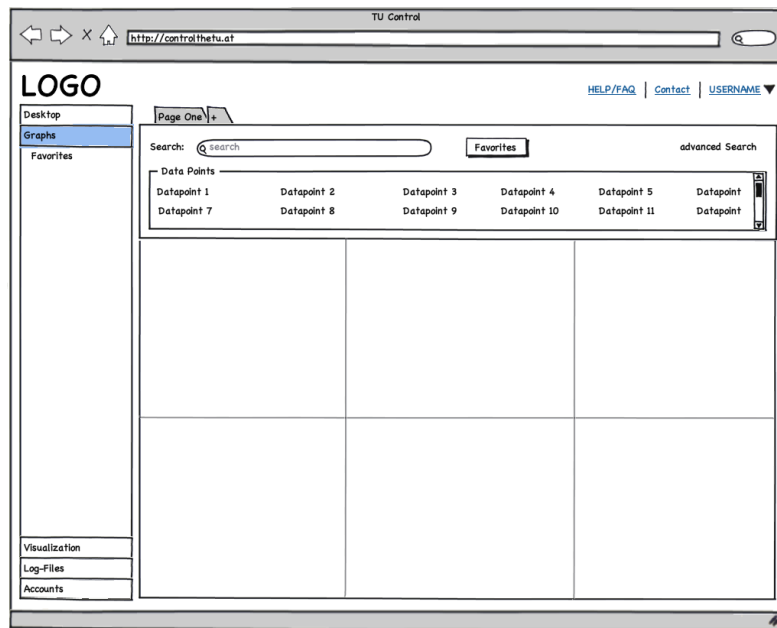
Every feature of the system is embed into a widget - window that is located in the main window. These windows can contain search functions, graphs, information or anything else. The widget with the global options and search of one menu-point is fixed at the top of the window, all others are customizable and can be scaled, moved and closed by the interactor. The scaling and moving is corresponding to the grid of the main-window.

The widget system provides a high flexibility for the users and for the developers. New features can be implemented without major changes in the system.





**Figure 4.8: UI Workflow Structure**



**Figure 4.9:** Mockup Graph Module: Stack Panel, Main Window (incl. grid and tabs), Fixed Widget (basic search view)

## Modules and Widgets

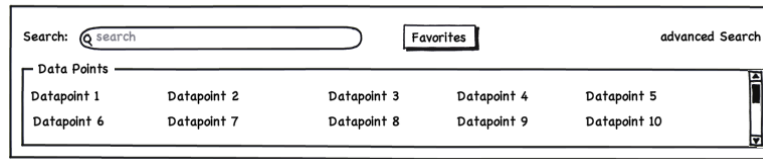
The modules “Desktop“, “Graphs“, “Visualization“, “Export“, “Log Files“ and “Accounts“ are quickly accessible via the stack panel menu (see Fig. 4.8) and cover all above described use cases (see “Use Cases“, page 25). The drag and drop function from the search function is integrated in the whole system. Widgets can be dragged and dropped from one to another module. For example a graph widget can easily be added to the Desktop by just dropping it in the stack panel on the menu point.

## Desktop

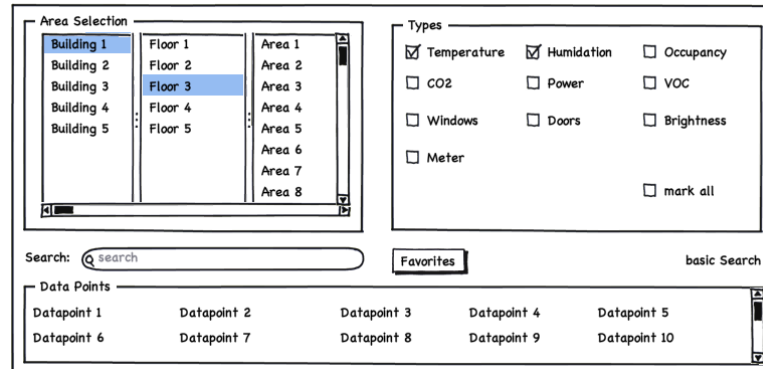
The “Desktop“ module is the overview of the interface. The option panel at the top of the main window allows to add standardized widgets and browse the favorites. The tab bar brings high flexibility for a custom sort order. This part is designed to be the “information center“ and additionally provides the data sets all notification and error messages.

## Standard Widgets

The system comes up with standard widgets for the “Desktop Module“ that cannot be personalized. In this matter the outdoor general weather conditions including a forecast



**Figure 4.10:** Search function: basic view



**Figure 4.11:** Search function: advanced view

or the notifications and log widget do not have any functionality than showing the actual values. Further widgets are “actual date and time“, the “RSS-Feed of the News-Blog“ or a “Calender“.

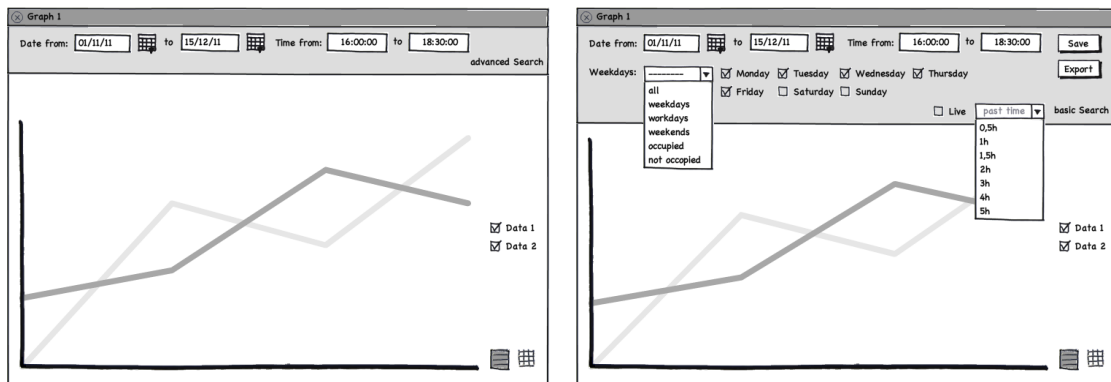
## Graphs

The “Graph“ module is intended to search for data sets and visualize them in a “Graph widget“. The search bar at the top of the main window is especially designed to quickly find the desired data point(s). In addition the “Favorite“ widget allows to browse through saved visualization settings.

### Search Function

The search function is a fixed widget and cannot be moved or closed. It is located at the top of the main window (see Fig. 4.8) and displays a search field and a list with all available data points (see Fig. 4.10).

An advanced view additionally includes a tab-browser for finding a certain area and the type definition of a sensor (e.g. Temperature, CO2, window, etc.) (see Fig. 4.11). Each limitation automatically reduces the number of data points in the list to find the desired one more efficiently.



**Figure 4.12:** Graph widget: basic / advanced view

Afterwards the data point is just dragged to the interaction window and a “graph widget” is generated automatically (Zach et al., 2012).

## Graph Widget

The graph widget is the most used and present one in the system. It consists of the graph itself and a search and limitation function (see Fig. 4.12). As already known from the standard search function this one also has got a basic and an advanced view with additional options.

The date and time range can be accessed all the time. Additional search criteria like days or predefined “special days” for example workdays or weekends can be accessed via extending the bar. The “Live” option allows to set a past time for example the last hour and the widget is updating automatically in real time.

To compare different data points, another one has just to be dragged from the standard search bar into the widget. At the right side of the graph all attached data sets are listed and can be activated / deactivated.

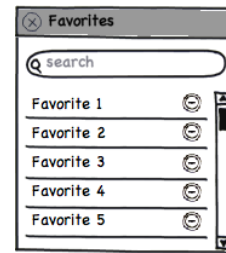
The graph is scale able via marking an area. This allows a quick analysis of a shorter period without using the search bar of the widget. Different grid styles help the user to optimize the current visualization.

The “export” button in the advanced view automatically copies the settings to the “Export Module”. Additionally the user can drag and drop the graph widget to the “export module” as well.

## Favorite Widget

The “favorite function“ allows to save settings of a data set and access it later again quickly. Each query can be named and it is available in a sub-level of the stack-panel menu or in the standard search bar.

This widget helps to save graphs that are not important to be visible all the time or to export a data set later on without redoing the search procedure.



**Figure 4.13:** Favorite widget

## Visualization

The “Visualization“ module contains a search function for browsing through the available 3D models. If there is a model of the whole building, every section or data point can be searched. The data points are dragged into the visualization and will be visualized immediately. The detailed adjustments of the values are adjusted inside the animation widget.

## Export

The “Export“ module contains a more advanced search function as already above described and allows an export to different file formats (see “Use Cases“, page 25). The data points are collected in a fixed export widget where further detailed settings (date, time, etc.) are possible.

## Log Files

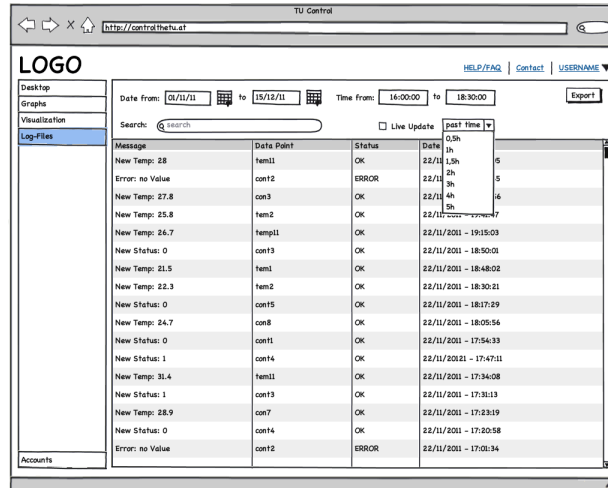
The “Log Files“ module (see Fig. 4.14) contains at the top a search function including date, time and live update. The logs can be browsed through and exported to csv and text files.

## Accounts

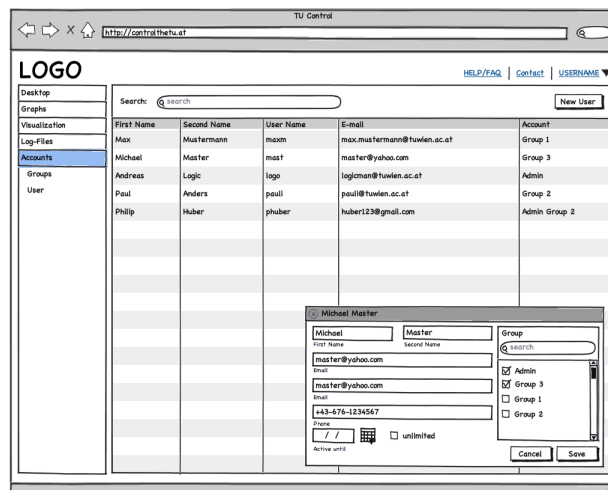
The “Account“ module (see Fig. 4.15) is structured into “Groups“ and “Users“. Each user has to be added to at least one group. A group regulates the accessibility to data points and functions.

An account can be limited for example to a certain area, data point types, or widgets or modules. Further on a user can get administration rights to create and administrate accounts. The option and search bar is situated at the top of the main window. It can be switched between group and account settings. Both have a search function to quickly

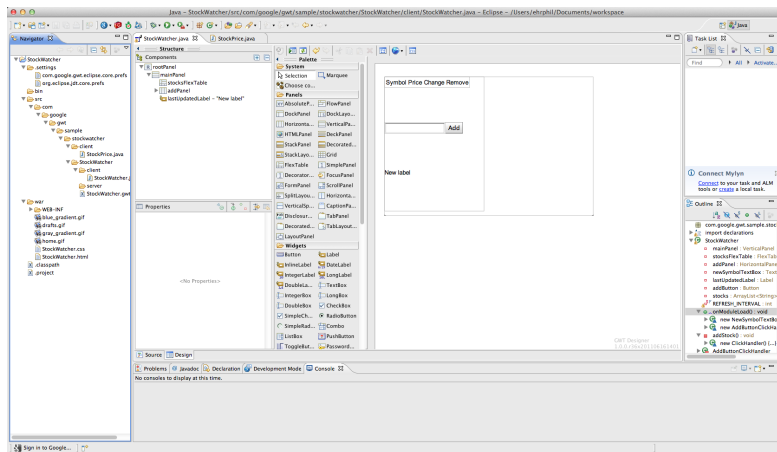
find the desired account or group. The editing or creating of a group or user is handled in an overlay window. This contains all text fields and settings in one level to grant a fast work flow.



**Figure 4.14:** Mockup Log Files Module: search-bar at the top and table of logs



**Figure 4.15:** Mockup Account Module: user administration, interface for editing and creating accounts



**Figure 4.16:** GWT in Eclipse | GWT Designer

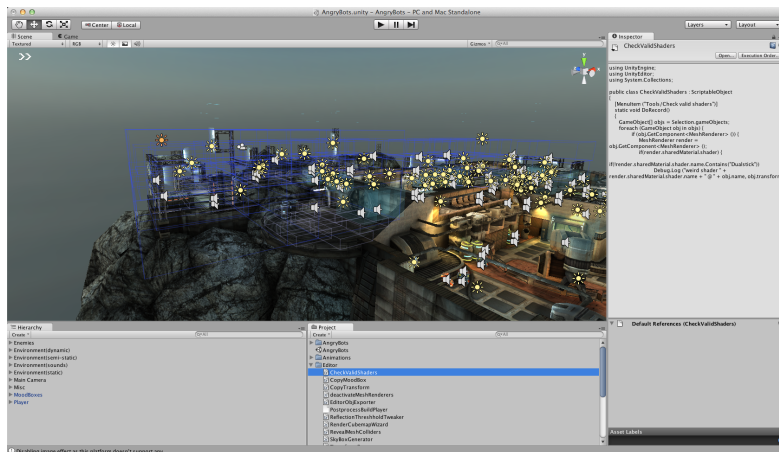
## 4.4 Technology

### Google Web Toolkit

Google Web Toolkit (GWT) is an open source<sup>1</sup> tool kit for web-engineering developed by Google Inc. released in 2006. The Java-to-JavaScript compiler (comp. Wikipedia, 2012b) allows a complete development of the client and server application in Java. The cross-compiler translates the Java-Code to standalone, optimized JavaScript files that run in every actual browser (Sowa et al., 2008).

JavaScript was the first popular browser based scripting language and is still the most popular one. Scripts are generally used to generate a function of HTML form input new HTML pages directly (Kappel et al., 2004). This speeds up a website because content can be loaded dynamically without loading the complete website for new. GWT is a plugin for the open source Java development framework “Eclipse“. The software development kit (SDK) can be extended with a bi-directional Java GUI designer. The “GWT designer“ (see Fig. 4.16) has got “What You See Is What You Get“ (WYSIWYG) (comp. Wikipedia, 2012a) layout tools and compiles the design automatically into Java without typing any line of code.

<sup>1</sup>Open Source Software (OSS) is free and in “source code form“ available. This free software is often developed in public, collaborative manner. Copyright is handled with special licenses like the “General Public License“ (GPL or GNU) and grants that modifications, redistribution and commercial use possible without paying for licensing to the original author. One of the most important open source projects is the web-server of the Apache Foundation. The system is used by more than 57% of the active websites for free in 2012.



**Figure 4.17:** Unity | Development Environment

## Unity

Unity is an authoring tool to create interactive content like video games, visualizations or real-time 3D animations developed by Unity Technologies. The development environment runs on Microsoft Windows and Apple Mac OSX. The produced interactive content can be compiled for Google Android, Apple Mac OSX, Apple Mac iOS (iPhone/iPad), Microsoft Windows, Microsoft Xbox360, Sony PlayStation 3 and Nintendo Wii. The Unity web player plugin is available for the browsers Mozilla Firefox, Apple Safari and Google Chrome on Mac OSX and MS Windows. Version 3.5 includes a Flash deployment add-on that allows to publish a project to the popular Adobe Flash<sup>2</sup> format.

The software is available as a free and pro version with additional features, such as global lightening or render-to-texture. Content produced with the free version is branded with a start screen or a water-mark in the web-player that cannot be customized or disabled.

## Work Flow

The 3D model is created and mapped in a 3D-modelling application such as Maxon Cinema4D, Autodesk 3ds Max, McNeel Rhinoceros or Google Sketch Up. The final model gets best imported to Unity as a .fbx or .3ds file including all textures.

<sup>2</sup>Adobe Flash is a commercial multimedia platform to display videos and animated or interactive content in web pages. The environment uses an object oriented programming language called “Action-Script“. There is no native support by any web browser. The free downloadable “Adobe Flash Player“ is available for several computer and mobile phone systems like Windows, Mac OSX, Linux or Android (Adobe, 2012)



Animating is handled quite simple via a record function. The framework supports three different scripting languages<sup>3</sup>, JavaScript, C#, and a dialect of Python named Boo. There is a list of predefined “event handler“ that start an action (script) i. e. “OnCollisionEnter“ starts an action if the defined object is entered. This list can be extended with personal ones as well. The component based architecture allows to attach a script to each object separately (Unity, 2012).

The coding does not differ to any other programming environment so an experienced programmer does not need a long time to understand the syntax. In addition the Unity asset online store offers extensions developed by the community, paid and free ones.

---

<sup>3</sup>Scripting versus Programming: These days the line between programming and scripting is blurred. In practical terms the difference is meaningless. Nevertheless it can be said that scripting is programming in a program. (Mischook, 2005)

## Prototype Evaluation

### 5.1 Objectives

An interactive 3D model of the department of “Building Physics and Building Ecology“ (see Floor plan A.2) of the Vienna University of Technology is developed as a prototype. The data visualization has to be accessible via a web browser, in the best case without requiring the installation of additional plugins (see “Use Cases“, page 25). The 3D-model visualizes real time data via colors, animations or just by showing the actual value (see Table 5.1). As already discussed the user interface has to be simple and quickly understandable.

The animation should help the user to identify problems or check the current status of an area remotely. This can save a lot of time for the facility managers and provides a higher level of service for the occupants. Further on, the data visualization should be understandable for the users without further introduction.

### 5.2 User Interface

The user interface of the 3D model (see Fig. 5.1) consists of just two main parts. The menu where the different data visualizations can be toggled on and off is located at top the right corner. In the diagonal axis, at bottom the left corner, the numeric values of “temperature“, “CO2 level“ and “relative humidity“ are situated. The actual temperature of an area is always displayed and cannot be turned off like all other values.

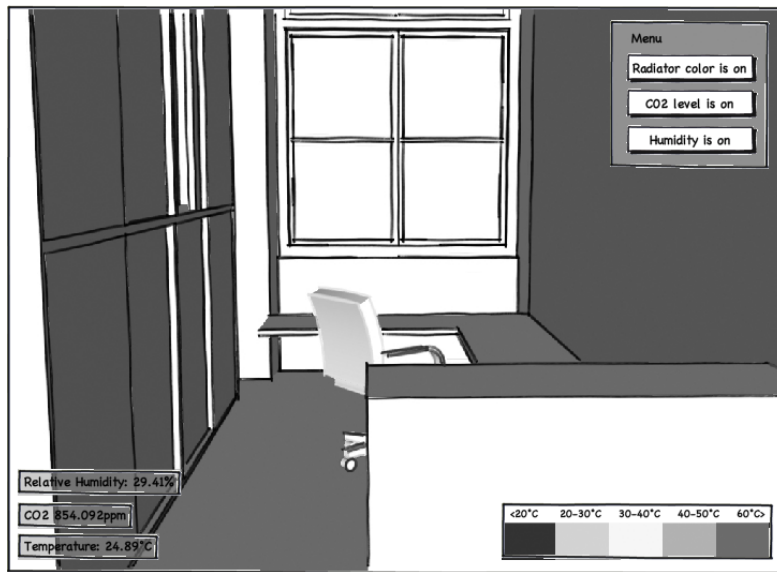
The radiator colorization activates a color-bar with the explanation of the corresponding temperature range (see Fig. 5.2). This part completes the UI in the right bottom corner. The interface is semitransparent and does not distract the user from the actual model and in-model data visualizations.

**Table 5.1:** Visualized real time data in the 3D model

Value	Visualization
Temperature of an area	The temperature is always displayed and updates automatically if the position is changed.
Temperature of the radiators	The temperature of the radiators is visualized via colors, depending on the degrees.
Air quality value	The CO2 percentage of an area can be displayed additionally as a value.
Status of windows and doors	The windows and doors automatically change their position in the model corresponding to their actual status.
Occupancy	An avatar is displayed if a room is occupied.
Illuminance	The status of a light is corresponding to the actual status (on/off).



**Figure 5.1:** User Interface of the 3D data model with all parts “Off”



**Figure 5.2:** User Interface of the 3D data model with all parts “On”

The navigation is handled with the arrow keys so the mouse is free to select the different menu options. The lock may not be familiar for users who are already used to navigate the directions with the mouse but it simplifies the learning process for new ones tremendously.

## 5.3 Data Visualization

The 3D model contains three different types of data visualizations of the monitoring network data, i) numerical, ii) status change and iii) colorization. Only the “main model”, outside and indoor walls and some cabinets have a “mesh collider”<sup>1</sup> attached to them and therefore prevent the user from walking through them.

### Numerical Visualization

The numerical visualization of data is used for temperature, CO2, and relative humidity as already above mentioned (see Fig. 5.4 and Fig. 5.5). These values do not require any other visual cues as they are well known by every user and can be assessed by personal experiences.

<sup>1</sup>The mesh collider in Unity builds its collision representation from the Mesh attached to the Object. This enables to define “physical” material properties.

The values are rounded to two places after the comma. More detailed values would just irritate the user and do not provide any further advantages in terms of assessment.

## Status Change Visualization

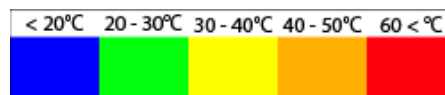
The status changes of doors, windows, occupants or lights can not be influenced by the user in the model. The animation for doors and windows is handled quite simple via displaying two different rotations of an object. To reduce distraction there is no animation of doors or windows opening or closing. If a status value changes, the closed door object is hidden and the 90 degrees turned door is displayed. For the occupancy status an avatar is displayed at the corresponding position of the sensor.

There is no impact in terms of navigation by the different status of doors and avatars. It is always possible to walk through them.

In terms of lights the user will mention bigger differences in the visualization. Shadows and reflections are altering in the cause of status changes.

## Color Visualization

The color visualization is a very simple but powerful tool to transmit information (comp. “Signals“, page 11). The five colors are corresponding to the temperature ranges (see Fig. 5.3) and allow the user to identify the actual status of a radiator quickly. The chosen bright colors differ substantially from all other objects in the model to identify each radiator quickly and from longer distances (see Fig. 5.5).

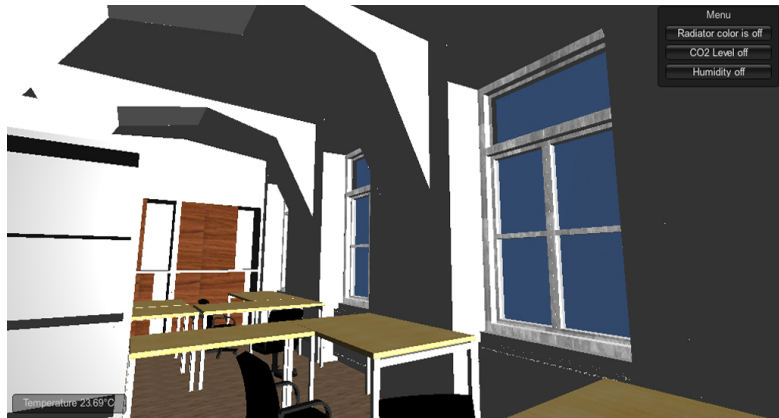


**Figure 5.3:** Color-bar of the radiator temperature ranges

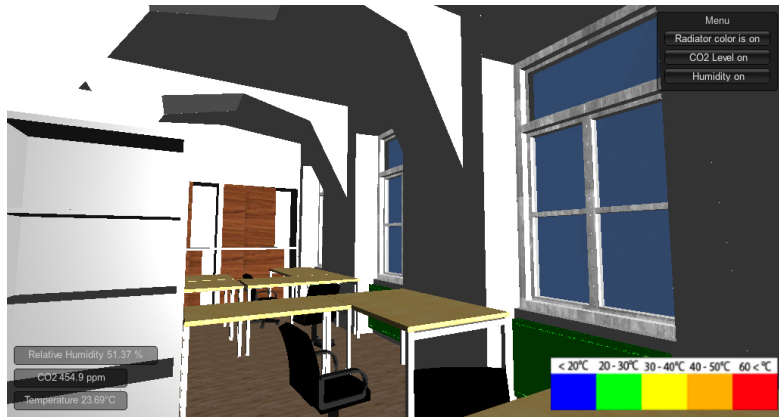
## 5.4 Data Model

The hierarchy consists of three levels, i) the “PHP connector“, ii) the “web access“ and “object controller script“ and iii) the “animation scripts“ (see Fig. 5.6). Inside the Unity environment JavaScript was used as programming language. It was not necessary to define any custom event-handler (comp. “Work Flow“, page 37) for the different visualizations and animations.

A PHP script (see Alg. B.1) accesses the MySQL database of the monitoring system as a standalone file. The file is hosted on a web server, and communicates with the “web



**Figure 5.4:** Screen shot of the real time data model with all parts “Off”



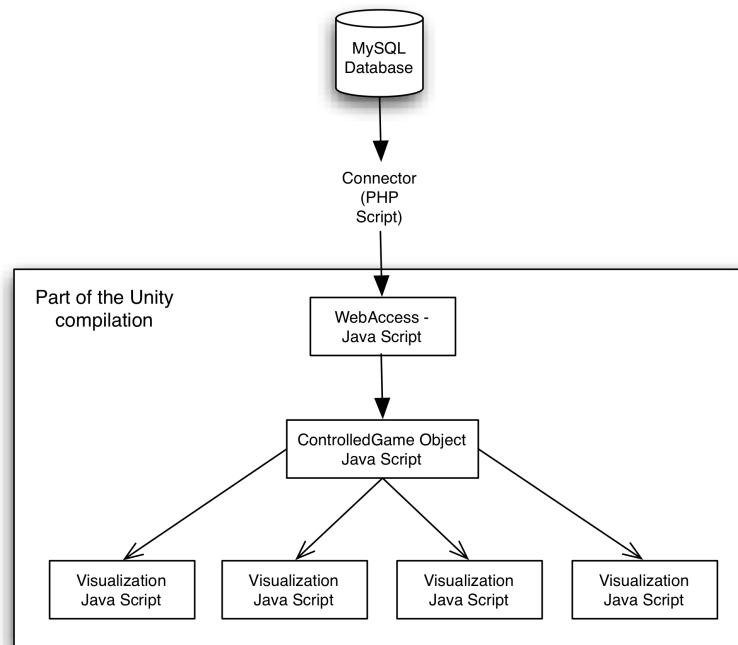
**Figure 5.5:** Screen shot of the real time data model with all parts “On”

access“ java script (see B.2) which is a part of the Unity model file.

Unity clients use this PHP script to query the database. On the one hand this keeps the database’s security credentials at a safe place. The credentials are never transferred over a public connection because the script is located on a server in the same local network as the database. On the other hand, this way the client never sends any query to the database directly but is only able to modify one parameter. These two security considerations prevent malicious or unauthorized operations on the database.

Having both, the database and the PHP script within the same local network also improves the queries’ speed significantly.

Each animated object of the model registers at the “Controlled Game Object“ (see Alg.



**Figure 5.6:** Data Connection Model: Database, Connector, Object Controller, Animation Scripts

B.3). The name of a mesh<sup>2</sup> is equal to the data point name in the database. This grants a logical object structure and high flexibility in terms of changes in the model.

The “valueChanged” function is implemented in each object and displays in the log every value change. Updates in terms of visualization are done every five minutes. There is no reason to create a high number of database queries to display minimal changes in the data model instant at this stage. There are other use cases that would require instant data visualization but these are not part of this work (see “Lessons Learned“, page 45).

The “Navigation Script“ (see Alg. B.5) is attached to the camera of the first person controller. It handles the menu and the color-bar (see Fig. 5.3) of the radiator colorization.

All other scripts are attached to each mesh with the data point name of the database. In case of the radiators the children of the mesh collection are addressed as well.

<sup>2</sup>A mesh [...] is a collection of vertices, edges and faces that defines the shape of a polyhedral object in 3D computer graphics and solid modelling. The faces usually consist of triangles, quadrilaterals or other simple convex polygons, since this simplifies rendering, but may also be composed of more general concave polygons, or polygons with holes (Wikipedia, 2012g).

## Conclusion

### 6.1 Lessons Learned

The visualization process in Google SketchUp is handled very simple and results in the desired model quickly. The export function provides the option of “two-sided” 3Ds files. This makes the handling of the normals of the faces obsolete and simplifies the import process.

The mapping in unity is similar to any other 3D-modelling application and supports all needed functions of transparency or specularly.

There is no support to change the “zero-point” of a mesh which is important in terms of animation, especially using rotation. This requires to import each desired animated object separately because SketchUp does not support multiple zero-points. More advanced modelling programs like Ciname4D or 3D Studio Max support this function.

Different lights like bulb or directional are provided by Unity natively, but there are limitations in the free version like missing live shadow rendering. Nevertheless, it was sufficient for this work.

The required “Unity web player” to display the model in a web browser is a quite high barrier for a wider range of accessibility. A compilation to the “Adobe Flash” format (see Unity, page 37) would reduce the barrier but still does not solve it.

Tests with interactive 3D visualization systems supported natively by the web browsers have shown major problems with navigation and rendering. These environments are at the beginning of their development and do not provide an authoring tool like Unity and have problems in terms of performance.

The “Web Graphics Library” (WebGL) has got the best opportunities to become a new and powerful interactive 3D graphic processing standard for web, but there is still a long way to go to reach the same performance level like Flash is already providing. WebGL



is an API for processing interactive 3D graphics in web browsers. The control code is written in JavaScript and the shadow processing is handled directly on the computer graphic processing unit (GPU). Several browser like Mozilla Firefox, Google Chrome or Safari as well as some mobile platform browsers do already support this technology (Wikipedia, 2012h).

An interactive 3D model like developed in this thesis, is not likely to become a standard in building monitoring systems, but it is a solid foundation for an augmented reality interface. Therefore further efforts in the area of motion tracking and specific hardware are indispensable to further improve the user experience. (comp. “New Ways, New Technologies“, page 17).

## 6.2 Future Research and Development

The actual 3D model only contains live data of the monitoring network. Especially for data simulations for example air ventilation or occupants statistics the model is a good base.

A connection to the “MATLAB”<sup>1</sup> environment would give access to the simulation data. The visualization of this data sets in the 3D model would not be possible with the standard event handlers and functions of Unity. Therefore new visualization scripts would be the only possibility to achieve high performance and flexibility.

Implementing an error notification system and a routing functionality that leads to a faulty sensor or any other malfunction and problem would require an instant data push of the database or at least a shorter interval of updating. Nevertheless, such a tool would provide a high benefit for facility managers and building maintainers.

At the end, as the intention of this work was to show new approaches in terms of building data visualization, it would be interesting to implement the model into an augmented reality system and start wider tests with facility managers about acceptance and usability.

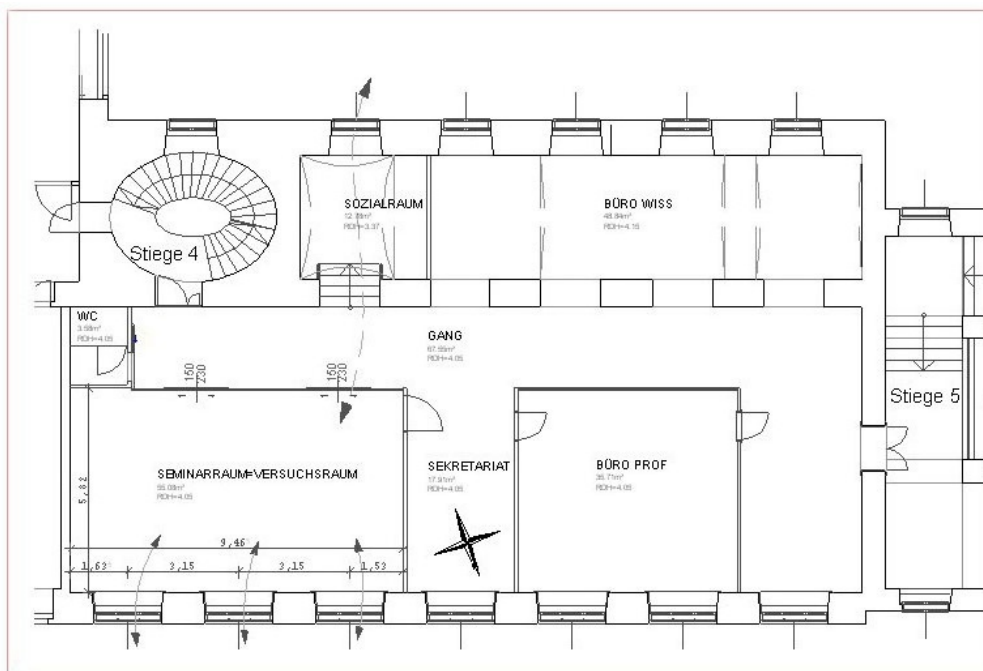
---

<sup>1</sup>“MATLAB” stands for “matrix laboratory” and is a programming environment for numeric computation. The system allows to design algorithm structures, do matrix manipulations or plot functions and data. Programs coded in Java, C, C++ or Fortran can interface with Matlab. The software was originally developed at the University of New Mexico. Today the development and licensing is handled by the company “Mathworks”.

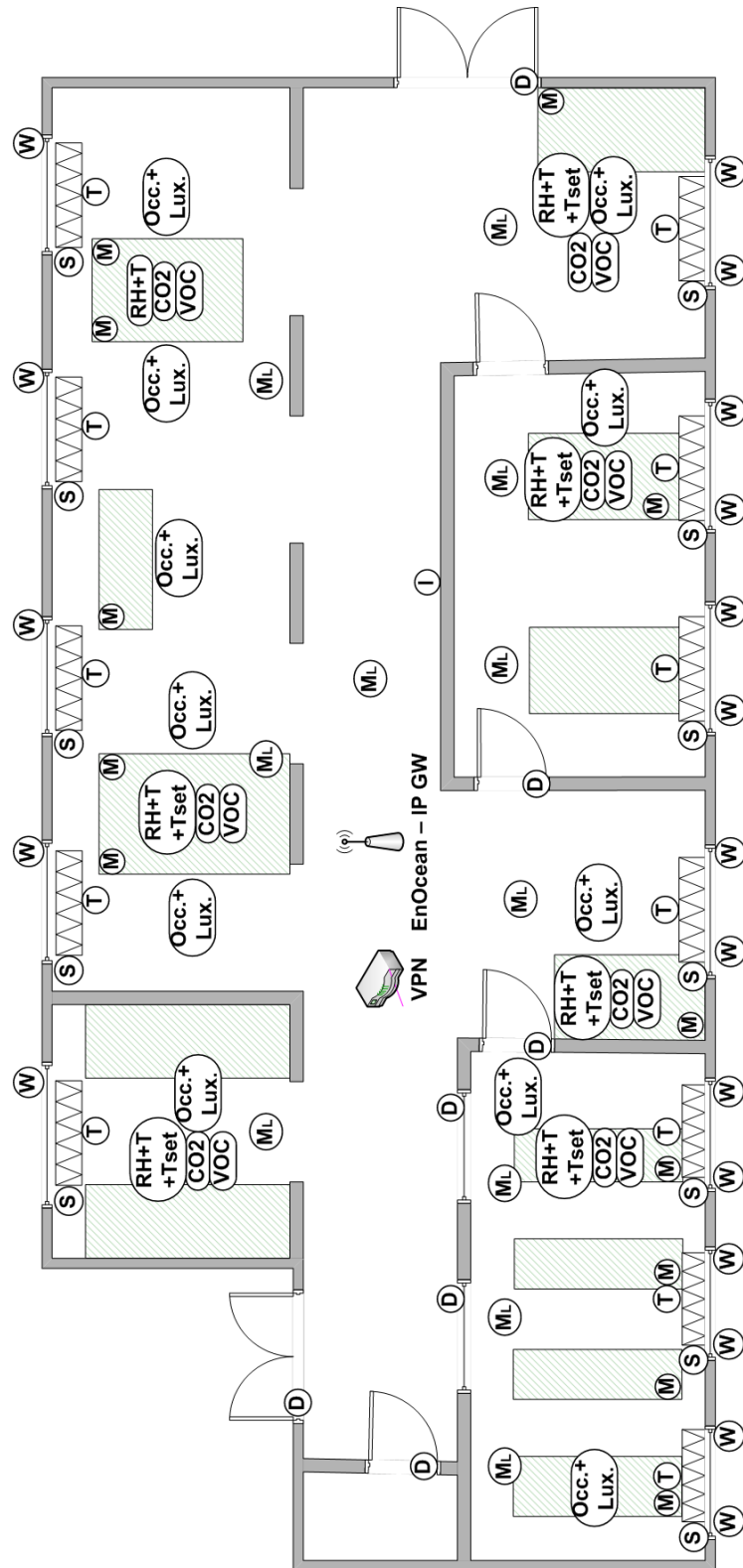
# APPENDIX A

## Floor Plans

### A.1 Department of Building Physics and Building Ecology



**Figure A.1:** Floor Plan “Department of Building Physics and Building Ecology“



**Figure A.2: BPI Sensors**

## A.2 Sensors

**Table A.1:** List of sensors and their acronyms

Sensor	Acronym
Carbon Dioxide	CO2
Door-contact	D
Information Monitor (Touch Screen)	I
Illuminance	Lux
Electrical meter	M
Occupancy	Occ
Relative Humidity	RH
Valve-actuator (Stellenantrieb)	S
Temperature-sensor	T
Temperature-setpoint	Tset
Volatile organic compound	VOC
Window-contact	W

# APPENDIX B

## Source Code

### B.1 PHP Source

```
1 <?php
2 $link = mysql_connect('host', 'user', 'password');
3 if (!$link) {
4     die('Could_not_connect:_ ' . mysql_error());
5 }
6 mysql_select_db('database_name');
7 $query = sprintf("select_value_from_data_where_
    datapoint_name_='_%'s'_order_by_timestamp_desc_limit_1",
8     mysql_real_escape_string($_GET['datapoint_name']));
9 $result = mysql_query($query);
10 if (!$result) {
11     $message = 'Invalid_query:_ ' . mysql_error() . "\n";
12     $message .= 'Whole_query:_ ' . $query;
13     die($message);
14 }
15 $row = mysql_fetch_assoc($result);
16 echo $row['value'];
17 mysql_close($link);
18 ?>
```

**Algorithm B.1:** PHP MySQL Access

## B.2 Unity Java Scripts

### Global Scripts

```
1 #pragma strict
2
3 private static var valueUrl = 'directory_of_' + "PHP_MySQL_"
   Access"_' ;
4
5 static function updateGameObjectValue(gameObject :
   ControlledGameObject) {
6     var url = valueUrl + '?datapoint_name=' +
       gameObject.name ;
7     var valueGet = WWW(url) ;
8     gameObject.valueWwwCallback(valueGet) ;
9 }
```

**Algorithm B.2:** Java Script: Web Access

```

1  #pragma strict
2  protected var value: float;
3  function Start () {
4      InvokeRepeating("updateValue", 0, 5); // update
        value every 5 minutes
5  }
6  function updateValue() {
7      Webservice.updateGameObjectValue(this);
8  }
9  function setValue(newValue: float) {
10     if (this.value != newValue) {
11         this.value = newValue;
12         valueChanged();
13     }
14 }
15 function valueWwwCallback(www: WWW) {
16     yield www; // process www request.
17
18     if(www.error) {
19         UnityEngine.Debug.LogError("There_was_an_error_
            getting_the_value_of_" + name + ":_ " + www.
            error);
20     } else {
21         try {
22             setValue(parseFloat(www.text));
23         }
24         catch (FormatException) {
25             UnityEngine.Debug.LogError("Could_not_
                parse_float_value_" + www.text + "_for_
                datapoint_" + name);
26         }
27     }
28 }
29 function valueChanged() {
30     UnityEngine.Debug.LogError("valueChanged_is_not_
        implemented!");
31 }

```

**Algorithm B.3:** Java Script: Controlled Game Object

## Visualization Scripts

```
1 #pragma strict
2
3 class Cdi extends ControlledGameObject {
4     var show_info = false;
5
6     function OnGUI () {
7         if (Navigation.co2 == true && show_info){
8             var roundedValue = Mathf.Round(
9                 this.value * 100) / 100.0;
10             GUI.Box ( new Rect (10,Screen.
11                 height - 65,150,25), GUIContent
12                 ("CO2_" + roundedValue + "_ppm
13                 "));
14         }
15     }
16
17     function OnTriggerEnter (other : Collider) {
18         show_info = true;
19     }
20
21     function OnTriggerExit () {
22         show_info = false;
23     }
24
25     function valueChanged() {
26         UnityEngine.Debug.Log("The_Room_" + name +
27             "_has_now_value_" + value + "!");
28     }
29 }
```

**Algorithm B.4:** Java Script: Numeric visualization; The values are rounded to two digits after the comma and can be turned on and off in the menu of the navigation interface (see Alg. B.5)



```

1  static var status = false;
2  static var co2 = false;
3  static var hum = false;
4  var colorrange : Texture2D;
5
6  function OnGUI () {
7      statusLabelRadiator = status ? "on" : "off";
8      statusLabelCO2 = co2 ? "on" : "off";
9      statusLabelHumidity = hum ? "on" : "off";
10
11     GUI.Box (new Rect (Screen.width - 170,10,160,105),
12             "Menu");
13     if (GUI.Button(new Rect(Screen.width -
14                             160,35,140,20), "Radiator_color_is_" +
15                             statusLabelRadiator)) {
16         status = !status;
17     }
18     if (status) {
19         GUI.DrawTexture (Rect (Screen.width - 310,
20                               Screen.height - 80, 300, 70),
21                           colorrange);
22     }
23     if (GUI.Button(new Rect(Screen.width -
24                             160,60,140,20), "CO2_Level_" + statusLabelCO2))
25     {
26         co2 = !co2;
27     }
28     if (GUI.Button(new Rect(Screen.width -
29                             160,85,140,20), "Humidity_" +
30                             statusLabelHumidity)) {
31         hum = !hum;
32     }
33 }

```

**Algorithm B.5:** Java Script: Navigation Interface including the menu for turning on and off the numerical visualization as well as the radiator colorization and the color-bar of the radiator temperature ranges (see Fig. 5.3).

```

1  #pragma strict
2  class Radiator extends ControlledGameObject {
3      function valueChanged() {
4          setRadiatorColor();
5          UnityEngine.Debug.Log("The_Radiator_ " +
              name + "_has_now_value_" + value + "!")
              ;
6      }
7      function getColorForValue() {
8          if (Navigation.status == false) {
9              return Color.white;
10         }
11         if (value < 20) {
12             return Color.blue;
13         }
14         else if (value < 30) {
15             return Color.green;
16         }
17         else if (value < 40) {
18             return Color.yellow;
19         }
20         else if (value < 50) {
21             return Color(1,0.5,0);
22         }
23         else {
24             return Color.red;
25         }
26     }
27     function setRadiatorColor() {
28         for (var child : Transform in transform) {
29             child.renderer.material.color = this.
                getColorForValue();
30         }
31     }
32     function OnGUI () {
33         setRadiatorColor();
34     }
35 }

```

**Algorithm B.6:** Java Script: Radiator Color

```

1  #pragma strict
2
3  class Displayfalse extends ControlledGameObject {
4      function Update () {
5          renderer.enabled = (value == 0);
6      }
7      function valueChanged() {
8          UnityEngine.Debug.Log("The_window/door_" +
9                                  name + "_has_now_value_" + value + "!"
10                                 );
11     }
12 }

```

**Algorithm B.7:** Java Script: Show / Hide Objects

# Bibliography

Adobe. Adobe flash, May 2012. URL <http://www.adobe.com>. last checked 21/05/2012.

Nick Bilton. Google to sell heads-up display glasses by year's end, February 2012a. URL <http://bits.blogs.nytimes.com/2012/02/21/google-to-sell-terminator-style-glasses-by-years-end>. last checked 20/03/2012.

Nick Bilton. Google begins testing its augmented-reality glasses, April 2012b. URL <http://bits.blogs.nytimes.com/2012/04/04/google-begins-testing-its-augmented-reality-glasses>. last checked 20/03/2012.

Bill Buxton. *Sketching User Experiences*. Elsevier Inc., San Francisco, USA, 2007.

Szu-Cheng Chien and Ardeshir Mahdavi. Evaluating interface design for user-system interaction media in buildings. *Advanced Engineering Informatics*, (22):page 484–492, April 2008.

Szu-Cheng Chien, Robert Zach, and Ardeshir Mahdavi. Developing user interfaces for monitoring systems in buildings. *IADIS International Conference Interfaces and Human Computer Interaction*, pages page 29–36, 2011.

Bruce Christie. *Human Factors of the User-System Interface*. Elsevier Science Publishers B.V., Amsterdam, Netherlands, 1985.

European Commission. Low energy buildings in europe: Current state of play, definitions and best practice. 2009.

Alan Cooper, Robert Reimann, and David Cronin. *About Face 3 - The Essentials of Interaction Design*. Wiley Publishing Inc., Indianapolis, USA, 2007.

red derStandard. Sergey brin trägt googles scifi-brille schon, April 2012. URL <http://derstandard.at/1333528528515>. last checked 20/03/2012.

DIN. Din en iso 9241-110, ergonomics of human-system interaction. 2006.

Richard Edwards, Ann Hansona, and Peter Raggatt. *Boundaries of Adult Learning*. Routledge, London, UK, 1996.

energiesparen-im-haushalt.de. Preisentwicklung öl und gas, January 2012. URL <http://www.energiesparen-im-haushalt.de/energie/bauen-und-modernisieren/modernisierung-haus/heizung-modernisieren/heizungsanlage-erneuern/preisentwicklung-oel-und-gas.html>. last checked 13/01/2012.

Vitaly Friedman. Data visualization and infographics. January 2008. URL <http://www.smashingmagazine.com/2008/01/14/monday-inspiration-data-visualization-and-infographics/>. last checked 01/03/2012.

Michael Friendly. Milestones in the history of thematic cartography, statistical graphics, and data visualization. August 2009. URL <http://www.math.yorku.ca/SCS/Gallery/milestone/milestone.pdf>. last checked 06/03/2012.

Wilbert O. Galitz. *The Essential Guide to User Interface Design - An Introduction to GUI Design Principles and Techniques*. Wiley Publishing Inc., Indianapolis, USA, third edition, 2007.

Google. Project glass: One day..., April 2012. URL [http://www.youtube.com/watch?feature=player\\_embedded&v=9c6W4CCU9M4](http://www.youtube.com/watch?feature=player_embedded&v=9c6W4CCU9M4). last checked 05/05/2012.

Gerhard Heufler. *DESIGN BASICS - From Ideas to Products*. Niggli Verlag AG, Zurich, Switzerland, 2004.

Gerti Kappel, Birgit Pröll, Siegfried Reich, and Werner Retschitzegger. *Web Engineering - Systematische Entwicklung von Web-Anwendungen*. dpunkt.verlag GmbH, Heidelberg, Germany, first edition, 2004.

Sami Karjalainen and Veijo Lappalainen. Integrated control and user interfaces for a space. *Building and Environment*, (46):page 938–944, June 2011.

Jens Lausten. Energy efficiency requirements in building codes, energy efficiency policies for new buildings. *International Energy Agency (IEA)*, March 2008.

Deana McDonagh-Philip and Anne Bruseberg. Using focus groups to support new product development. *Institution of Engineering Designers Journal*, September 2009.

- Stefan Mischook. Scripting vs. programming: is there a difference?, September 2005. URL <http://www.killersites.com/blog/2005/scripting-vs-programming-is-there-a-difference>. last checked 22/04/2012.
- Christina Portalés, José Luis Lerma, and Santiago Navarro. Augmented reality and photogrammetry: A synergy to visualize physical and virtual city environments. *ISPRS Journal of Photogrammetry and Remote Sensing*, 65(1):page 134–142, January 2010.
- Jennifer Preece, Yvonne Rogers, and Helen Sharp. *Interaction Design - Beyond Human Computer Interaction*. John Wiley & Sons Inc., New York, USA, 2002.
- Paul Raftery and Marcus Keane. Visualizing patterns in building performance data. *Proceedings of Building Simulation 2011: 12th Conference of International Building Performance Simulation Association*, Sydney, Australia, November 2011.
- Hans Sowa, Wolfgang Radinger, and Martin Marinschek. *Google Web Toolkit - Anjax Anwendungen einfach und schnell entwickeln*. dpunkt Verlag GmbH, Heidelberg, Germany, first edition, 2008.
- Jenifer Tidwell. *Designing Interfaces - Patterns for Effective Interaction Design*. O'Reilly Media Inc., Sebastopol, USA, 2006.
- Edward R. Tufte. *The Visual Display of Quantitative Information*. Graphics Press, Cheshire, USA, second edition, 2001.
- Unity. Authoring tool, February 2012. URL <http://www.unity3d.com/>. last checked 20/02/2012.
- Fernanda Viegas and Martin Wattenberg. How to make data look sexy. April 2011. URL [http://articles.cnn.com/2011-04-19/opinion/sexy.data\\_1\\_visualization-21st-century-engagement?\\_s=PM:OPINION](http://articles.cnn.com/2011-04-19/opinion/sexy.data_1_visualization-21st-century-engagement?_s=PM:OPINION). last checked 01/03/2012.
- Wikipedia. Wysiwyg - what you see is what you get, February 2012a. URL <http://de.wikipedia.org/wiki/WYSIWYG>. last checked 15/02/2012.
- Wikipedia. Java (programming language), February 2012b. URL [http://en.wikipedia.org/wiki/Java\\_\(programming\\_language\)](http://en.wikipedia.org/wiki/Java_(programming_language)). last checked 14/02/2012.
- Wikipedia. Application programming interface (api), May 2012c. URL [http://en.wikipedia.org/wiki/Application\\_programming\\_interface](http://en.wikipedia.org/wiki/Application_programming_interface). last checked 13/05/2012.

Wikipedia. Augmented reality, April 2012d. URL [http://en.wikipedia.org/wiki/Augmented\\_reality](http://en.wikipedia.org/wiki/Augmented_reality). last checked 20/03/2012.

Wikipedia. Command line interface (cli), January 2012e. URL [http://en.wikipedia.org/wiki/Command-line\\_interface](http://en.wikipedia.org/wiki/Command-line_interface). last checked 12/01/2012.

Wikipedia. Graphical user interface (gui), January 2012f. URL [http://en.wikipedia.org/wiki/Graphical\\_user\\_interface](http://en.wikipedia.org/wiki/Graphical_user_interface). last checked 12/01/2012.

Wikipedia. Polygon mesh, May 2012g. URL [http://en.wikipedia.org/wiki/Polygon\\_mesh](http://en.wikipedia.org/wiki/Polygon_mesh). last checked 19/05/2012.

Wikipedia. Web graphics library (webgl), May 2012h. URL <http://en.wikipedia.org/wiki/WebGL>. last checked 22/05/2012.

Wikitude. Augmented reality browser, April 2012. URL <http://www.wikitude.com>. last checked 22/02/2012.

Robert Zach and Ardeshir Mahdavi. Monitoring for simulation validation. *BauSim*, September 2010.

Robert Zach, Stefan Glawischnig, Regina Appel, Johannes Weber, and Ardeshir Mahdavi. Building data visualization using the open-source most framework and the google web toolkit. April 2012.