FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

AN URBAN MONITORING SYSTEM FOR LARGE-SCALE BUILDING ENERGY ASSESSMENT

DISSERTATION

zur Erlangung des akademischen Grades

Doktor der technischen Wissenschaften

eingereicht von

Dipl.-Ing. Stefan Glawischnig, BSc
Matrikelnummer 0803925

an der

Fakultät für Informatik der Technischen Universität Wien
Betreuung: Univ.Prof. Dipl.-Ing. Dr.techn. Ardeshir Mahdavi

Diese Dissertation haben begutachtet:

(Univ.Prof. Ardeshir Mahdavi) (Univ.Prof. Schahram Dustdar)

Wien, 29. Februar 2016

(Stefan Glawischnig)

FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

AN URBAN MONITORING SYSTEM FOR LARGE-SCALE BUILDING ENERGY ASSESSMENT

DISSERTATION

submitted in partial fulfillment of the requirements for the degree of

Doktor der technischen Wissenschaften

by

Dipl.-Ing. Stefan Glawischnig, BSc
Registration Number 0803925

to the

Faculty of Informatics at the Vienna University of Technology
Advisor: Univ.Prof. Dipl.-Ing. Dr.techn. Ardeshir Mahdavi

The dissertation has been reviewed by:

(Univ.Prof. Ardeshir Mahdavi) (Univ.Prof. Schahram Dustdar)

Wien, Monday 29th February, 2016

(Stefan Glawischnig)

Erklärung zur Verfassung der Arbeit

Dipl.-Ing. Stefan Glawischnig, BSc
Meidlinger Hauptstraße 42-44, Top 71/72, 1120 Wien

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.

(Wien, 29. Februar 2016) (Stefan Glawischnig)

Acknowledgements

At this point I would like to express my gratitude towards my colleagues, friends and family. Above all, I want to thank my supervisor Professor A. Mahdavi, who guided my first steps in the scientific community. Our talks influenced me not only on a professional level, but also personally. Furthermore, I would like to thank Professor S. Dustdar for his willingness to officiate as my second examiner.

I would like to thank my parents, Hans and Irmgard, who greatly influenced me throughout my entire life. They always did everything they could to support me and my education. Thanks Michi and Kathi for being there for me. Without my family I would not be who I am today.

Thanks to all my colleagues at the BPI for their input and support. Thanks Harald and Christian for your amazing efforts in our research projects.

Besides all the positive events, a PhD naturally comes with a good portion of frustration at times, overnight stays in the lab and tons of fast food. I want to thank Marissa for her countless efforts to improve my eating habits (I became a huge fan of avocados) and her last-minute review of my dissertation.

Everyone that has not explicitly been mentioned - thank you! Last but not least, I would like to congratulate myself - pretty good job.

Abstract

Building a generic data structure that handles building related data at an urban scale offers certain challenges. Real world entities must be captured in an environment that allows for the communication of relevant data. This thesis deals with the development of an urban monitoring and simulation framework to investigate building performance and energy demand on a larger scale. An effort is described that aims to enhance a well tested building monitoring system to handle building data at an urban scale. This requires the development of a distributed, generic and enhanceable data store, as well as the conceptualization of a modular and scalable application architecture. The scalable data store is introduced, as well as the modularization process of the application logic, including data handling and communication routines. Beside handling monitoring data, the potential that open government GIS-data offers for the proposed toolkit is investigated. Two-dimensional GIS-data is used to derive simplified geometric building models that can be used to calculate standardized building energy demands. A method that utilizes this geographic building models to automatically assess the respective energy demands according to the ISO 13790:2008 standard is introduced. Beside the calculated energy demands, a building product ontology is presented that is used to calculate retrofit scenarios for the respective buildings. A prototypical implementation shows how the toolkit can be used based on the example of approximately 900 buildings in Vienna. The building model generation process, an implementation of the proposed energy demand assessment method, and the retrofit calculation method are discussed in detail.

Kurzfassung

Die Konzeptualisierung einer generischen Datenstruktur zur Bearbeitung von gebäudebezogenen Daten auf urbaner Ebene bietet gewisse Herausforderungen. Komplexe Objekte der realen Welt müssen erfasst und in eine Umgebung einbezogen werden, die die Kommunikation relevanter Daten erlaubt. Die vorliegende Arbeit beschäftigt sich mit der Entwicklung eines urbanen Monitoring- und Simulationtoolkits, dass zur Untersuchung von Gebäudeperformance und Energiebedarf auf urbanen Maßstäben dienen kann. Es wird beschrieben, wie ein getestetes Gebäudemonitoringsystem erweitert wird, um Gebäudedaten auf urbanem Maßtab zu erfassen. Dies setzt einerseits die Entwicklung eines verteilten, generischen und leicht erweiterbaren Datenspeichers, andererseits die Konzeptualisierung einer modularen und skalierbaren Applikationsarchitektur voraus. Der Datenspeicher wird vorgestellt, als auch der Modularisierungsprozess der Applikationslogik, inklusive Datenverarbeitung und Datenkommunikationsroutinen. Neben monitoringbezogener Arbeiten wird das Potential von Open Government Geodaten für das vorgestellte Toolkit untersucht. Zwei-dimensionale GIS-Daten werden verwendet um vereinfachte geometrische Gebäudemodelle zu generieren. Diese Gebäudemodelle können zur standardisierten Berechnung des prognostizierten Energieverbrauchs verwendet werden. Eine Methode, um ebendiesen anhand der geographischen Gebäudemodelle entsprechend des ISO 13790:2008 Standards automatisiert zu berechnen, wird vorgestellt. Aufbauend auf dieser Methode wird eine Ontologie von Gebäudeprodukten eingesetzt um Sanierungszenarien für die respektiven Gebäude zu berechnen. Eine prototypische Implementierung zeigt am Beispiel von ungefähr 900 Gebäuden in Wien, wie das Toolkit eingesetzt werden kann.

Der Prozess, indem die geographischen Gebäudemodelle generiert werden, die Implementierung der vorgestellten Methode zur Abschätzung des Energiebedarfs und die Methode zur Berechnung der Sanierungsszenarien werden im Detail vorgestellt.

Contents

CHAPTER 1

Introduction

1.1 Introduction

Urbanization and sustainability strategies (Europe 2020) raise the need to address ecological and economic issues on an urban level. The enormous global challenges that are associated with energy crises, climate change, accelerated resource depletion, and growing population raise the need to optimize the consumption of energy, reduce the waste of resources and stop the increasing environmental pollution (IPCC 2014). A comprehensive view of urban features allows investigating large-scale phenomena, such as the analysis of the energetic behavior of entire neighborhoods (Glawischnig et al. 2014a, Glawischnig et al. 2014b, Kiesel 2015).

Besides the large-scale observation of building behavior and performance, the efficient and accurate distribution of resources in an urban environment, such as loads in supplying grids is of interest. Furthermore, a comprehensive view on energy demands in urban settings offers possibilities for urban planning and socio-economic decision making regarding the development of funding instruments. As the building sector is responsible for up to 40% of the overall energy usage in many countries (Graubner and Hüske 2003) , buildings should not be recognized as stand-alone entities but as parts of a dynamic system that interact with their surroundings (Mahdavi 2012).

A comprehensive observation of building behavior and performance can offer insight into the complex ecological and economic dependencies and influences on political, administrative, socio-economic, technical, and planning domains that are concerned with building performance. The urban view point and the variety of connected fields nurture the emergence of new research fields. For instance, the optimization of resource distribution in urban environments, such as loads in supplying grids and the consideration of local renewable potential is of interest.

Currently, most buildings in the existing building stock provide yearly energy demand profiles. High-resolution data is sparsely available and data collection is often limited by data privacy regulations. This high-resolution data can be generated and maintained by monitoring systems (Building Monitoring Systems BMS, Energy Management and Control Systems EMCS, Smart Meters) that unfortunately often lack interoperability. The department of Building Physics and Building Ecology, TU Wien developed a platform and vendor independent, Java based BMS that follows a distributed and service oriented approach (Zach and Mahdavi 2012, Zach 2012, Zach et al. 2012, Zach et al. 2013b).

The urban environment produces massive amounts of data, not all of them directly related to buildings, but nevertheless influential. For instance, it is common practice to integrate weather forecasts and historical weather data into a building's EMCS as well as to utilize it as base data for light and thermal simulation (Schuss et al. 2013, Saipi et al. 2013). Urban decision support systems depend on (i) weather data, (ii) data on traffic networks and density, (iii) demographic and social factors, (iv) building morphology, (v) building types (e.g. living spaces, business areas, etc.), (vi) occupancy patterns, and many more. Comprehensive decision support can help the involved stakeholders (e.g. energy providers, construction companies, legal authorities, urban planners, end users, etc.) in their decision finding processes.

This work presents an effort to create a platform independent urban monitoring toolkit that allows collecting and processing multiple building data streams (e.g. sensor data) simultaneously. A lightweight conceptual

framework to predict and estimate energy demands of buildings on an urban scale is developed upon the proposed toolkit. The assessed energy demands are then used to predict retrofit costs with the help of a building product ontology. This is achieved by utilizing GIS-data that is available through open data interfaces. Similar to the united states, the European Union generally mandates the implementation of open data interfaces. Administrative data is made publicly accessible to increase transparency and to support participation and collaboration.

It is described how open government data is processed to derive a lightweight geographic Building Information Model (gBIM) that (i) offers all properties that are necessary to calculate heating and ventilation losses, (ii) allows the exchange of multiple building models simultaneously due to reduced file sizes and to (iii) support common thermal simulation software (e.g. EnergyPlus - UIUC and LBNL (2007)). The automatically generated gBIM models are used to calculate building energy demands with simplified geometries and standardized assessment methods.

These simplified energy demand assessments are calculated according to the ISO 13790:2008 *Energy performance of buildings - Calculation of energy use for space heating and cooling* (ISO 2012) standard. The national implementation of this standard for Austria, which is published as ÖNORM B8110-6 (ASI 2014) and ÖNORM B8110-5 (ASI 2011) is used to create a lightweight mathematical model for the calculation of GIS-based energy certificates. Due to the two-dimensional nature of the available GIS-data and the lack of certain attributes (e.g. wall materials, retrofit documentation, roof geometries, etc.), the mathematical model of the standard is simplified to handle GIS-input data.

For this purpose, a solid data basis must be established that (i) is able to store and provide building data for multiple buildings and (ii) automatically generate all the necessary semantic data to allow the usage of the gBIM models to calculate energy demands. Based on Open Government Data a procedure is introduced that derives all the necessary geometric and attributive information needed to run basic energy demand calculations.

1.2 Motivation

The work in this paper focuses on the conceptual development and prototypical implementation of a multi-purpose environment to process and visualize complex energetic processes in the built environment. To depict large-scale models from real world scenarios, various data sources must be taken into consideration. These are (i) real-time monitoring data, (ii) building geometries (iii) building attributive data and (iv) sensor data regarding urban conditions (e.g. weather data).

Previous research within this domain concentrated on the development of the building monitoring system MOST (MOST 2012, Zach et al. 2012, Zach and Mahdavi 2012). Buildings are represented by topological models, which consist of datapoints and zones and omit physical properties and geometries. Datapoints are any kind of data producing entities (e.g. sensors, users). Zones are administrative entities that allow managing datapoints (for instance rooms or building floors) hierarchically. The goal was to develop a storage concept that handles building related data in a unified way, regardless of the initial data format or producer, to help removing data-related disincentives and to facilitate the integration of building performance evaluation in building design and retrofit (Wolosiuk et al. 2013). This work documents an effort to adjust and extend the proposed building-monitoring framework MOST to handle building data at an urban level. The goal is to develop a conceptual model that allows establishing a connection between building monitoring systems and various other data sources to support urban monitoring and simulation scenarios. As it can be seen in Figure 1.1 these scenarios comprise

- **Real-time Urban Monitoring**

  Data feeds from an arbitrary number of buildings or sensors can be integrated into one environment to be used for centralized data analysis and visualization, as well as for further data distribution. The infrastructure to store and process diverse building data partly originates from the MOST environment and is transferred to an urban domain.

- **GIS-based energy demand assessment**

  Based on spatial geometry data, topologically correct, low-detail geographic building models (gBIM) are automatically generated to offer a starting point for the development of simplified thermal building models. Open data sources are used to equip these building models with attributive information to implement a low-detail mathematical framework for standardized energy demand assessment.

- **Computation of retrofit scenarios**

  Based on user specified boundaries (selected buildings, time frame) the mathematical model that is used for the heating demand calculation is utilized to run retrofit scenarios. An ontology that contains building product data is used to specify the retrofit material and to determine the retrofit costs.

- **Simulation and real-time calibration**

  The integrated dynamic monitoring data feeds and static building models are used to develop interfaces to third party simulation tools. These simulations (e.g. energy demand) are constantly updated with real-time monitoring data to establish calibrated thermal building models. An demonstration of this concept was implemented at building level by Tauber et al. (2014).

- **Web-based urban BIM applications**

  Beside the automated integration of simulation tools, human interaction with the data and models should be integrated into the workflow. This could be building architects that verify BIM with monitoring data or simplified/ detailed thermal building models. BIM exchange and collaborative BIM are also use cases that can build upon the proposed platform.

The USIM (Urban Simulation and Monitoring Toolkit) introduced in this work partly utilizes concepts of MOST. The urban data structure partly depends on the topological MOST data model (datapoints organized by zones). Furthermore, data processing routines and data distribution concepts are party adapted to serve the need of generic urban data handling

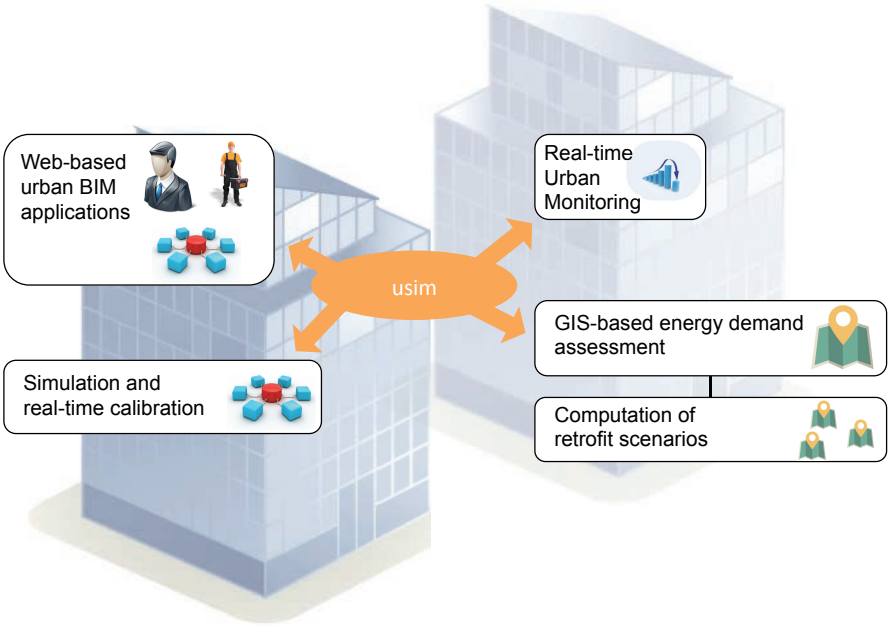

Figure 1.1: Urban monitoring and simulation scenarios

and analysis. Beside the integration of building monitoring data a variety of other data sources that do not address internal building related phenomena, such as climate settings, are integrated into the environment.

Assessing influences on buildings in general and urban energy performance in particular requires a variety of data streams. This can for instance be weather data that is collected by research institutions, private individuals or data that is hosted by governmental institutions and is accessed via services (e.g. open government data), or demographic data, traffic data, building morphologies, etc.

The European Union generally mandates the implementation of open data interfaces. In 2007, the Directive 2007/2/EC of the European Parliament and of the Council (EU 2007) established a infrastructure for Spatial Information in the European Community (Parliament and European Council 2015). INSPIRE defines 34 spatial data themes that are of interest for environmental applications. The INSPIRE directive aims to establish rules for spatial data exchange between the member states. On a national level, administrative data must be made publicly accessible to increase transparency and to support participation and collaboration.

To illustrate the potential of the proposed monitoring structure, a GIS-

application is introduced that utilizes the monitoring framework to assess energy use on district level. Urban scale microclimatic simulation requires complex and time consuming numeric calculations (e.g. computational fluid dynamics). As to the high complexity of the real world, finding a feasible set of boundary conditions that suits generic urban analysis with numeric simulations is hardly possible. A generic simulation model that suits a variety of urban research fields is hardly definable. Therefore, simpler methods to describe and estimate energy demands on an urban scale are needed. Simplified mathematical models and available energy performance data are used to calculate energy use within a sample district in Vienna. These calculations include a variety of factors, e.g.

- Building monitoring data
- Weather data
- Physical properties (e.g. materials)
- Geometric data (e.g. building outline, height).

Beside this use case, an urban data collection offers the possibility to support various research efforts. The assessment of passive and active renewable potential at a larger (urban) scale requires a variety of data streams, methodologies and tools. To support stakeholders and decision makers regarding energy efficiency matters in urban planning processes, certain requirements must be fulfilled. These requirements depend on:

- Information on spatio-temporal intensity distribution of renewable supply potential (solar, wind, geothermics)
- Information on spatio-temporal intensity distribution of energy demand on an urban level (HVAC, light, machinery)
- A systematic comparison of energy supply and demand to deliver a basis for decision finding in energetic planing processes.

To integrate continuous simulation and monitoring applications into the proposed structure, the concept of virtual datapoints (Zach et al.

2013b) can be adapted. Virtual sensors entail data representations that are not bound to physical sensors. For instance, this might be the on-demand calculation of an entire building's accumulated energy consumption, in case no smart meter data is available. Furthermore, the storage concepts and the application architecture are revised to address urban related requirements (e.g. performance optimization, conceptual adjustments, etc.).

Beside an revision of existing MOST code bases and their extension and optimization, other tools and methods that support the mentioned urban research scenarios are analyzed and integrated. For instance, experience regarding building products and ontologies that was gathered in research projects offers a starting point for the implementation of the urban retrofit scenarios module and a building product ontology.

1.3 Research Statement

The following section entails the starting point of the research:

1. A generic data storage and processing mechanism allows the simultaneous data handling of a variety of urban entities (e.g. buildings, zones, sensors, etc.).

   The implementation of a distributed, high-performance data store allows the storage of building data from multiple buildings. Decoupling data preprocessing and analytical functionality from the data storage engine allows to simultaneously process building applications.

2. Decentralized urban data handling can support a variety of use cases for urban research topics.

   The data communication via topological models and standardized service interfaces provides the basis for the implementation of a variety of urban research topics.

3. Vendor and technology independent urban building monitoring is of central importance for future proof, domain comprehensive data

collections. Building related data is available in a variety of formats and differing quality. Vendor independent interfaces and data communication routines are vital to support future urban research fields.

4. Urban monitoring applications can support collaborations of a variety of stake holders. The integration of a variety of data streams offers the possibility to support a variety of stake holders in their decision finding processes (e.g. smart grid applications, energy providers, planning of ecological and economic instruments, subsidies, etc.).

5. Open government data interfaces provide valuable input for urban research topics. Beside the solely building related data streams (e.g. sensors), publicly accessible data (e.g. building geometries from cadastres, statistical data regarding population, etc.) provide new valuable input for urban research topics.

6. Integrating multiple building related data sources, allows to build a multi-building urban monitoring system. The concept of an urban monitoring system requires the integration of various data feeds. This includes static data such as geometry models and dynamic data such as sensor measurements in various temporal resolutions.

7. Linking data sources of various types to buildings. The existing topological model (zones, datapoints, virtual datapoints) that is used to describe buildings is revised to be transferred to an urban level. Well structured building related data must be linked to a simplified geometric building model that is derived from GIS data.

1.4 Aim of the Work

The focus of this work is on developing an urban monitoring and simulation toolkit, which is accomplished by:

1. The implementation of a data storage concept that is capable of storing monitoring data of multiple buildings and provides sufficient access performance.
2. Automatically generate building models from GIS-data.
3. Develop a lightweight method to assess energy demands on an urban level, by including various data sources and catalogs.
4. Use the developed concepts to compute retrofit scenarios for a building sample.
5. Demonstrate the feasibility of the proposed methods.

1.5 Related Work

The following section intends to give a short overview (completeness not guaranteed) of ongoing research efforts at our university.

1.5.1 MOST

This work builds upon the platform and vendor independent toolkit MOST that was thoroughly documented by Zach et al. (2012), and is discussed in more detail in Chapter 2. MOST was developed within the scope of a research project, that was funded by the the Austrian Science Foundation (FWF): Project: I 545-N22 (Ubiquitous dynamic building performance monitoring) until 2012.

1.5.2 SEMERGY

The Semantic Technologies for Energy-efficient Building Planning (SEMERGY) project that was conducted between 2011 and 2015 investigated the potential benefits that semantic data mining technologies could offer to building performance assessment applications. Within this project, a complex BIM, a detailed method to calculate heating demand, ecologic performance and investment costs for building retrofit were developed. The basis for the optimization approach to calculate retrofit scenarios that

was implemented within the framework rests on a product ontology. This ontology "intends to bridge the gap by providing semantic links between real world products and building model's abstract concepts and elements" (Mahdavi et al. 2015, pp. 364). To realize a comprehensive data collection, the Baubook web platform (Baubuch 2013) was used as a main resource for product data. The SEMERGY framework is thoroughly documented by Pont et al. (2015), Pont (2014), Fenz et al. (2014), and Pont and Mahdavi (2015).

1.5.3 E_PROFIL

E_PROFIL is a set of methods (an IT-supported toolkit) for the elaboration of neighborhood profiles. The aim of the project is to facilitate an energy and resource efficient development in the planning practice of Austrian cities. Furthermore, the project is an important asset for research and planning activities in Europe and can also be applied to other neighborhoods. E_PROFIL is a research project that started in 10/15, funded by the Austrian Research Promotion Agency (FFG 2015). The methods introduced in this paper will most likely be build upon in this project.

1.5.4 Internet of Things

There are a number of ongoing Internet of Things (IoT) related research efforts. Yao et al. (2015a) introduced a case study about a smart home monitoring system based on a Web of Things (WoT) platform. This system focuses on the as accurate as possible monitoring of human behavior with the goal to analyze human physical activities and develop decision support in everyday life. Sensors and sensing activities are exposed as services and resources via a web-based interface. The prototype has been implemented in inhabited home environments.

Gatica (Qanbari et al. 2015) is a middleware for IoT gateways that enriches raw sensor data using annotations and transforms the data into RDF (Beckett 2004) objects and streams these objects to an analytic query endpoint that directly allows to query data by sending SPARQL (Harris and Seaborne 2013) requests.

Jung et al. (2014) developed an integration middleware for IoT. It provides a comprehensive communication stack for embedded devices based on IPv6, web services and oBIX with the aim to provide interoperable, generic interfaces for smart objects. Further information can be found in Ziegler et al. (2013) and Jung (2015).

Buildings must be considered as a part of an interactive urban system. The incorporation of IoT into urban cloud applications in smart cities has been thoroughly discussed by Dustdar et al. (2014). They address the challenges that working with domain-specific, tightly coupled systems provided by specific vendors offer in the context of smart cities.

1.5.5 SIMULTAN

SIMULTAN (2016) is an ongoing research project that deals with the multidisciplinary planning processes of refurbishment projects and new developments in highly efficient cities. It aims to increase energy efficiency to realize resilient and sustainably cities. This is done by developing a method to create a typologisation of the energy usage in a city to derive scenarios for the development of energy supply.

1.6 Structure of the work

The thesis is structured in terms of 7 Chapters, including the introduction (Chapter 1).

- **Chapter 2: State of the Art.**

  This chapter gives an overview of the necessary background that is needed to place the following chapters in context to building scientific research, such as Big Data collections in buildings and in urban settings. Beside a general introduction, previous work of the author that stands in line with this thesis, such as an overview of the building monitoring system MOST, and the development of GIS-based algorithms to assess the urban morphology, is introduced.

- **Chapter 3: From Building Monitoring to Urban Monitoring.**

Chapter three deals with monitoring related developments. The proposed building monitoring system will be introduced, as well as the efforts to enhance the existing infrastructure to build a distributed, generic data store and application architecture for urban monitoring. The core modules were refactored and processing logic separated from the data store. Beside the conceptual developments, the actual implementation, including the application's core components will be discussed (for instance virtual datapoints).

- **Chapter 4: GIS-based Energy Demand Assessment.**

  Chapter four describes the development of a GIS-based energy demand assessment method that utilizes the proposed monitoring structure to simulate energy use on district level. Furthermore, it covers how GIS-geometries are used to automatically create geographic building models. The standard mathematical model for calculating energy certificates is simplified to be applied to the geographic building models that are characterized by simplified geometries. This building models are used to assess the energy demand for buildings according to current standards.

- **Chapter 5: Urban retrofit scenarios.**

  Based on the methods from Chapter 4, retrofit scenarios will be computed for a collection of buildings. The methods (e.g. program sequence) and the mathematical model to calculate the retrofit scenarios are introduced. Beside the business logic an ontology, which contains building products that can be used for the calculation of retrofit scenarios, is described.

- **Chapter 6: Prototypical Implementation.**

  This part describes a prototypical implementation of the methods developed in this work. At the example of a web-application, the modules that prototypically implement the concepts presented in this thesis are introduced. The functionality is visualized by application screenshots.

- **Chapter 7: Conclusion.** The last chapter discusses implications of the presented work and an outlook to possible future research. It furthermore includes a list of the contributions that are relevant for the thesis, co-authored with Prof. Mahdavi and project-team members and colleagues.

CHAPTER 2

State of the Art

2.1 Big Data in the Built Environment

The term *Big Data* emerged as an abstract definition for data collections that cannot be processed by traditional data analysis methods. These collections are characterized by a lack of structure, dynamic behavior and rapid growth. Handling rapidly growing unstructured datasets raised the question on how to classify and process big data to derive an economic value. The origin of the term Big Data is rather unclear, however, in 2001, Gartner developed a definition that described the aforementioned attributes of big data as part of large scale business process definitions (Laney 2001). Following the definition Gartner (2001) proposed, big data is defined by three characteristics:

- **Volume:** Development of strategies to handle growing data sources. This contains methods to reduce unused or redundant data, outsourcing of data warehouses, priorization and monitoring of data usage.
- **Velocity:** Technical innovations, such as fiberglass or hardware improvements consistently increase possible data access speeds and availability as possible competitive differentiators. Data handling strategies must be developed with a focus on the minimization

of the latency between end users and processes of the business logic.

- **Variety:** The biggest challenge is to handle various, partly unstructured data formats and to offer data exchange and translation mechanisms between inconsistent and often incompatible data sources.

Rapidly growing data collections can support quantitative studies. However, drawing conclusions from big linked datasets can only deliver correlations. This originates from the circumstance that most of these data stores are not created to fit a certain purpose or research question. As Mayer-Schonberger and Cukier (2013) criticise, big data analyses are often used do derive a causality without considering the fuzziness and correlative nature of the results:

"Most strinkingly, society will need to shed some of its obsession for causality in exchange for simple correlations: not knowing why but only what" (Mayer-Schonberger and Cukier 2013, pp. 7). This aspect of big data collections must be considered when utilizing such data for research purposes. Shifting the research focus on the act of collecting data might result in a certain degree of fuzzyness in the actual data sets as the focus is removed from data structure and nature.

As Big Data is typically associated with large, global operating companies and governments in the context of data mining activities, a growing level of digitalization in building automation technologies introduced the need to handle rapidly growing building-related amounts of data. Klein et al. (2013) summarize big data definitions and research challenges. At the example of twitter, youtube and facebook, the authors introduce state of the art technologies such as NoSQL databases and parallel processing techniques like Apache Hadoop (Apache 2015b).

These web-based digital technologies led to a utilization of building data to document and monitor the behavior of buildings. Building specific data collections support the development of a variety of applications that support building operation, such as building monitoring, predictive control

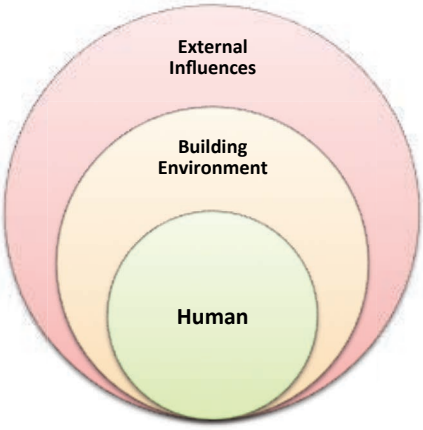

Figure 2.1: Influences on building data.

and simulation, and building commissioning. As it is demonstrated in Figure 2.1, within the context of this work, big building data is divided into three categories:

- **Data directly related to human activities:** Buildings are objects constructed to support human activities. As such, building users have an impact on building operation. Data related to human activities include occupancy monitoring, energy usage (for instance from electrical appliances, artificial light sources, etc.), manual heating and cooling and the resulting temperatures and CO_2 levels.

- **Automated building environment:** Certain parts of building technology that are not directly adjustable and influenced by occupants define a frame for stable operation. These centrally controlled mechanisms can be the control of Heating, Ventilation and Air Conditioning (HVAC) and automated internal and external shading elements as well as the control of lighting systems.

- **Environmental Influences:** Beside the importance of the human impact on the building performance, environmental data such as the influences on heating and cooling demand of global solar radiation, weather and the outdoor climate in general highly influence both building and occupant performance.

There are various ongoing research efforts in different disciplines that aim to investigate human behaviour at different scales. In times of mobile devices and social networks, human activities offer the possibility to develop context-aware systems that adapt according to changes in human behavior. In 2007, Baldauf et al. introduced a conceptually layered framework for representing, storing and exchanging contextual information. The work presented in Chapter 3 partly aims to follow the presented approaches on a building level.

2.2 Building Monitoring

Building Monitoring Systems (BMS) offer functionality to store, process, distribute and visualize building related data for audiences with varying professional backgrounds, such as visitors, users, facility management, building management, technicians, energy providers, etc. Beside manual analysis, interfaces for automated data processing should be provided. The goal of BMSs is to offer comprehensive building data collections and data processing functionality with appropriate access times.

Growing numbers of installed sensors and improved architectures of monitoring systems are used to develop real time applications. Real time, high-denisity sensor networks can be used to optimize building performance through fault detection. Furthermore, fine-grained sensoric building representations can be used to derive accurate occupancy patterns to support building automation. Khan and Hornbæk (2011) investigate a number of real-time occupancy-centric applications that use different sensors to monitor user behavior. The project Dasher (Autodesk 2015) utilizes high-resolution LIDAR models to generate BIMs that are used to visualize occupancy data.

Handling growing data collections requires optimized IT-Infrastructure. For instance, new databases that are more suitable to store highly dynamic sensor data emerged. Project Dasher utilizes Folio, a Internet-of-Things database system, that was optimized for real-time building automation (SkyFoundry 2015). Within this work, the Key-Value datastore

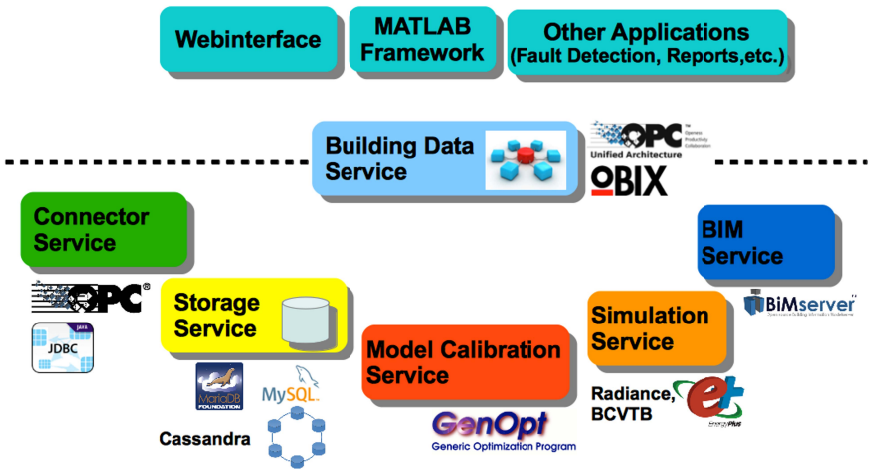

Figure 2.2: MOST Components (Zach et al. 2013b).

Cassandra (Apache 2015a) and the graph database Neo4j (Neo4j 2015) are used as storage mechanisms for sensor measurements.

2.3 Monitoring System Toolkit (MOST)

This work builds upon MOST that was developed within the scope of recent research efforts at the Department of Building Physics and Building Ecology of the TU Wien, to provide a vendor and platform independent building monitoring framework. The following chapter intends to give a concise overview of the core concepts to provide a context for the work presented later on. The work discussed in this chapter has been thoroughly documented within the scope of various research papers, specifically by Zach (2012), Glawischnig et al. (2012), and Glawischnig and Mahdavi (2013). MOST intends to provide a platform for (i) processing, (ii) retrieving, (iii) persisting, (iv) accessing, and (v) presenting building related data (MOST 2012). It follows a distributed and service oriented approach (Zach et al. 2013b). As it can be seen in Figure 2.2, the toolkit consists of six services. Conceptually, buildings are represented by a topological model that consists of datapoints and zones.

Datapoints are data producing entities within a building context. These are mainly sensors, but also persons can be considered as datapoints. Each person has a unique perception of the environment. This perception

can be used to monitor user generated measurements. These measurements are manually enterd by the user (e.g. through a web page). Zones are administrative entities that provide toplogical contexts for datapoints.

$$\forall sensors \in building, \exists zone : sensor \in zone \tag{2.1}$$

As Equation 2.1 shows, each building contains an arbitrary number of zones wherein each datapoint belongs to at least one zone. A zone can refer to entire buildings, floors, rooms and (sub-)spaces accross/ inside rooms. Moving to an urban scale extends the set of possible zonal objects. These are zones that extend beyond multiple buildings, building blocks, districts and other urban areas.

Datapoints refer to exactly one physical sensor. At times, it might be beneficial to group various sensors, introduce standardized transformations (e.g. $^{\circ}C$ to $^{\circ}F$) or to observe physical phenomena that are not directly accessible (e.g. the accumulated energy use of all datapoints inside a specific zone). For this purpose virtual sensors were developed. A virtual sensor is a stand-alone program that works at a relatively low application level and thus behaves like a native datapoint. Beside data manipulation, virtual datapoints can be used to extend the application's core functionality, and are extensively used to support simulation, model calibration, and data prediction.

Currently, virtual datapoints are implementations of an abstract `Datapoint` class and are identified via a unique ID (String). As it can be seen in Figure 2.3, datapoint implementations are managed by the `DpController` that returns a reference to the respective datapoint implementation that is requested via the unique identifier. This concept is discussed in more detail by Zach (2012).

2.3.1 Data Storage

The storage service is currently defined as the central datastore, not as the persistance layer of the framework. The data store uses a relational database (MySQL) to (i) store (ii) access and (iii) process data. As it can

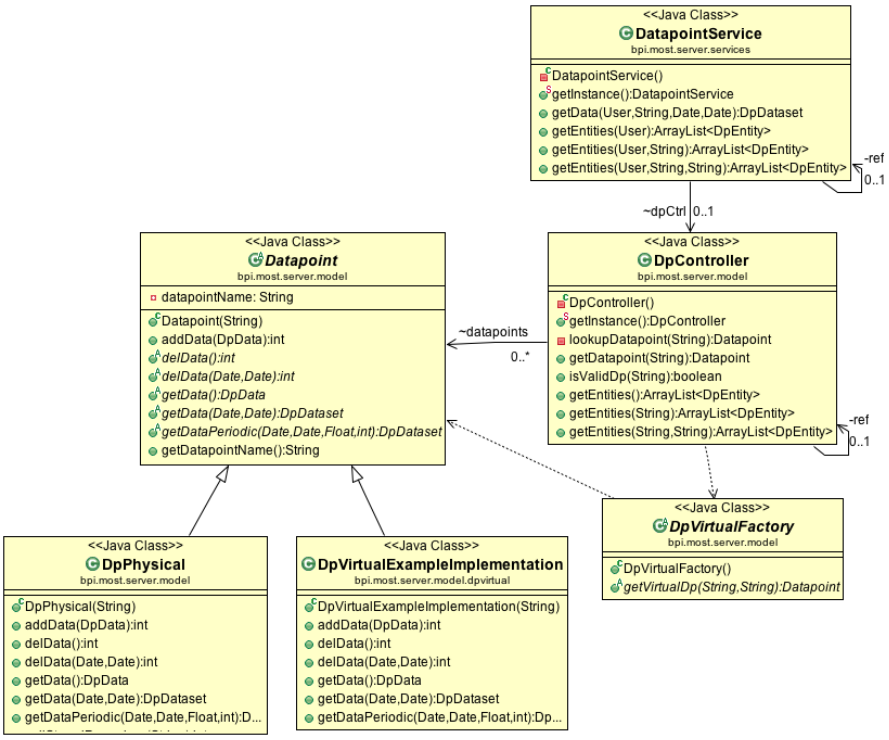

Figure 2.3: Virtual datapoints class diagram (Zach 2012)

be seen in the ER-Diagram (Figure 2.4), sensor measurements (`data` table) are defined by three attributes:

`<datapoint_name, timestamp, value>`

Sensor metadata, such as recording time intervals, sensor type, unit, name and description are stored in the `datapoint` table. Data access operations (read, write, update) are encapsulated in stored procedures (Zach et al. 2012). The benefit of using stored procedures is that all possible queries are maintained in the database schema. However, changing the processing logic requires an update of the entire schema. The procedures access the respective tables (i.e. `data, datapoint, zone` to read configuration data and write new measurements. An overview of the currently implemented procedures is provided by Table 2.1. There are three possible access types: (i) read *r*, (ii) write *w* and (iii) a combination of read and write (rw) that is either used in `UPDATE` statements or more advanced processing logic.

Table 2.1: Stored procedures with access type (r = read, w = write)

Procedure name	Type
addData(dp, ts, value)	rw
addDataForced(dp, ts, value)	w
emptyDatapointTimeslot(dp, start, end)	rw
calcAverageWeighted(dp, start, end, start value)	r
emptyDatapoint(dp)	rw
getNumberOfValues(dp)	r
getValues(dp, start, end)	r
getValuesPeriodic(dp, start, end, period, mode)	r
getValuesPeriodicAnalog(dp, start, end, period, mode)	r
getValuesPeriodicBinary(dp, start, end, period, mode)	r
getValuesPeriodicWhereDpBetween(dp1, start, end, period, dp2, valueLow, valueHigh, modeDp1, modeDp2)	r
getValuesPeriodicWhereDpBigger(dp1, start, end, period, dp2, value, modeDp1, modeDp2)	r
getValuesPeriodicWhereDpEqual(dp1, start, end, period, dp2, value, modeDp1, modeDp2)	r
getValuesWhereDpLower(dp1, start, end, period, dp2, value, modeDp1, modeDp2)	r
getValuesWhereDpBetween(dp1, start, end, dp2, valueLow, valueHigh)	r
getValuesWhereDpBigger(dp1, start, end, dp2, value)	r
getValuesWhereDpEqual(dp1, start, end, dp2, value)	r
getValuesWhereDpLower(dp1, start, end, dp2, value)	r
interpolateValuesLinear(dp, start, end, period, value start, value end)	r

For instance, new sensor measurements are tested by the respective stored procedure for sensor specific constraint violation. These constraints can refer to sensor specific recording time intervals, value ranges, filtering of measurements by time ranges or comparisons between selected datasets. Due to reasons of availability (e.g. empty battery, solar powered, measuring intervals, network availability) and type (e.g. event-driven measurements) sensor data can be collected at irregular intervals. To provide regularly distributed data series, data querying supports interpolation algorithms for periodic data generation (for instance *getValuesPeriodic(), getValuesPeriodicAnalog()*). These stored procedures load measurements into the database memory and interpolate between real measurements to generate periodic datasets. This is done by applying methods such as calculating time-weighted averages, linear interpola-

tions or majority decisions between values. The implemented algorithms are documented in more detail by Zach et al. (2013a).

2.3.2 Data Access Interfaces

Data producing entities (e.g. sensors, or static weather files) are mapped to datapoints in the database. The connection is established via customizable `Connectors`. Connectors are programs that map measurements to datapoint values via a specified driver (for instance JDBC, OPC DA). Connectors are instantiated via the Java ServiceLoader. This allows adding new sensors during runtime. At startup, every connector instance connects to the database and requests configuration data of the linked datapoint. Beside the communication with sensors, connectors can be used to push larger datasets periodically to the database (for instance weather files from a weather station).

Besides communicating with the database directly via stored procedures within the application context, data access is provided via two standard industry protocols, OPC Unified Architecture (OPC UA) and Open Building Information Xchange (oBIX). Additionally, a custom RESTful and RPC service are integrated. Currently, within the services the implemented resources are (i) zones, (ii) subzones, (iii) datapoints, and (iv) data with methods for reading and writing values to the desired objects. Data communication relies on a valid `User` object. To ensure a maintainable design, data access is handled by an `EntityManager`. Specifically, datapoint related queries are generated by the `DatapointFinder` and data related queries by the `DpDataFinder`.

2.3.3 Client Applications

The toolkit contains two standard applications that offer user access to the BMS. The web application was designed with a specific focus on usability (Glawischnig 2013), and is divided into various modules that fit specific use cases, such as (i) managing datapoints, (ii) creating graphs, (iii) exporting data, (iv) providing feedback and (v) providing a multidimensional building representation.

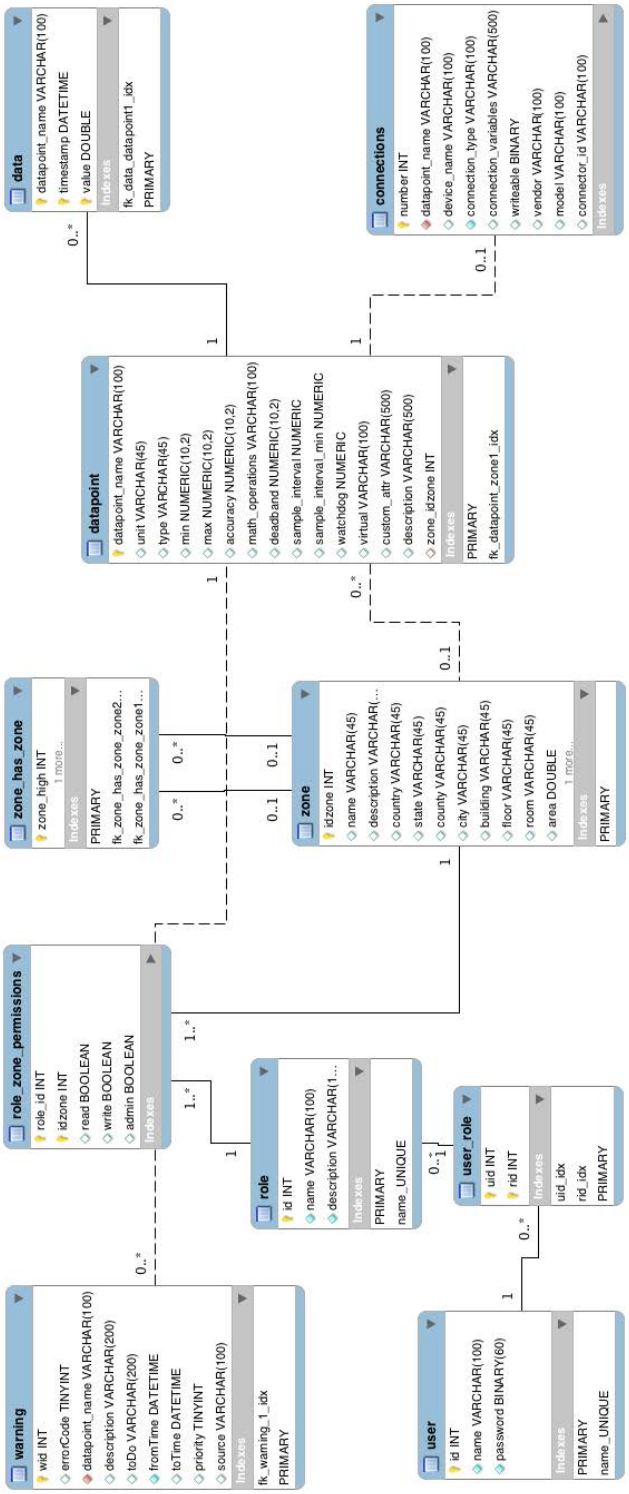

Figure 2.4: Relational database schema from MOST.

Within a web based environment it is possible to interact with live building data within a three dimensional building model (Figure 2.5). Datapoints are represented as objects within an Industry Foundation Classes-model (IFC) and are mapped to the respective database entity via a unique ID. Default map navigation operations are translated into a three dimensional context to lead the user within a building model. The IFC-model is converted to a sceneJS scene that is displayed by using a customized version of BIMsurfer (BIMsurfer 2014). SceneJS is a WebGL-based engine that is based on Javascript and JSON models. BIMsurfer is an open source web-based viewer for the visualisation of IFC/BIM models that is either using sceneJS input (via a JSON file) or models that are stored in a object relational database (BIMserver 2014). The functionality of BIMsurfer was enhanced with functionality to expose building storeys, and to change the transparency of wall elements. Equipment within rooms is equipped with a sensor ID, that is used to communicate sensor measurements from the database to the building model.

For instance, chairs are linked to occupancy sensors, doors and windows to contact sensors, lights to sensors that measure the intensity of solar radiation, and electrical equipment to sensors that measure the amount of used electrical energy. The intention behind this development is to provide users with an intuitive way to query real-time sensor data.

2.3.4 Calibration, Simulation and Building Information Models

Within the scope of virtual datapoint implementations, a concept for automated simulation model calibration was developed. As it has been discussed by Tauber et al. (2014, pp. 265) monitored data is used to "... evaluate and calibrate simulation models to improve the predictive potency of calibrated models". User-related monitoring data (e.g. usage schedules) is used to calibrate thermal building simulation models (Weber et al. 2012). Within a virtual datapoint GenOpt is started. GenOpt is an optimization program for the minimization of a cost function (GenOpt 2014). This cost function is evaluated by an external simulation program.

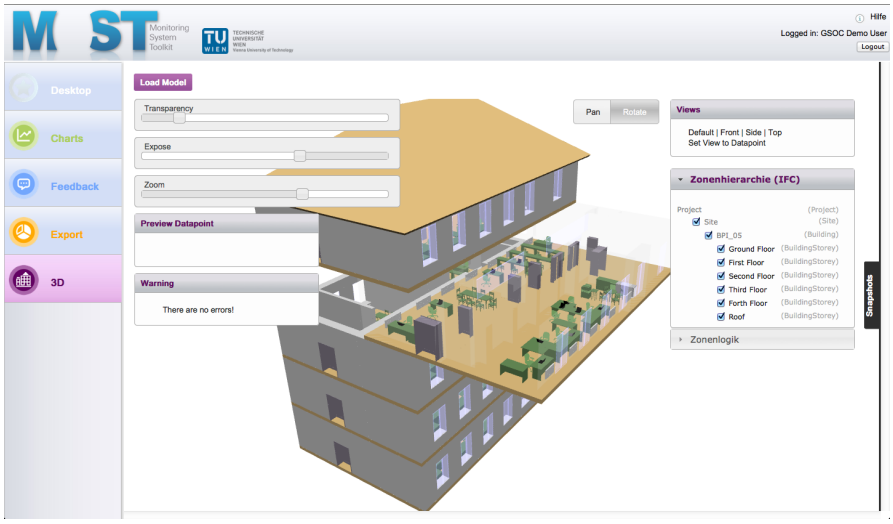

Figure 2.5: Web application: interactive 3-dimensional building model. The models are displayed by a customized version of BIMsurfer.

Currently, EnergyPlus support is integrated for thermal simulation.

To expose GenOpt functionality within MOST, a GenOpt plugin extension was developed. This way, the EnergyPlus simulation is initialized by GenOpt, but the cost function is calculated within the monitoring framework. This process can be summarized by the following steps:

1. GenOpt starts an EnergyPlus simulation within a virtual datapoint.
2. The simulation results are returned to the virtual datapoint.
3. The calibration service requests monitored data for the simulated zones.
4. The simulation results and monitored data are used to calculate the cost function.
5. This result is used to update the simulation input data for the next optimization step.

There are various possibilities to implement building models. A purpose of BIMs is to incorporate attributive properties of builings as well as the geometry to the software aided support of building planning and operation. Depending on use case specifications, various standards can

be used to generate BIMs, for instance IFC, CityGML or gbXML. The previously introduced client applications depend on IFC-models.

The BIM service intends to provide building geometries for the three dimensional building viewer application as well as an alternative to store datapoint configuration data. Within a BMS life cycle, sensor locations and definitions change frequently. Beside the opportunity to manually enter or update sensor definitions within the database or application, building models could be used to store sensor configurations. However, these features are still being developed.

2.4 GIS-based analysis of Open Government Data

Urban development is often characterized by changes introduced in land use, which are followed by the evolution of complex spatial structures in metropolitan areas. These distinctive urban features influence the microclimate by altering the total energy balance of the city and often implicate higher temperatures due to heat storage in urban areas (Nunez and Oke 1977, Grimmond and Oke 1999).

This circumstance, referred to as the urban heat island (UHI) phenomenon (Oke 1982), has been the focus of a number of studies, for instance by Giridharan et al. (2004) and Unger (2004). The urban climate is affected by a variety of influences such as geometry and vegetation (Memon et al. 2007, Santamouris 2007).

Increase in average temperatures has far-reaching implications that wield influence on environmental properties such as thermal comfort (heat stress), mortality, energy consumption and urban planning (Taha 1997, Lee and Park 2008).

In this context, a systematic framework for the assessment of microclimatic extremes in large-scale spaces was developed by Mahdavi et al. (2013) and Kiesel et al. (2013). Various geometric properties and topological relationships are used to calculate microclimatic attributes for five representative areas in the city of Vienna.

The following section describes the development process of geometry-

based spatial algorithms that use vector and high-resolution raster data of Vienna to calculate (i) sky view factor, (ii) aspect ratio, (iii) pervious and impervious surfaces, (iv) built and unbuilt area fraction, (v) mean building compactness, (vi) built surface fraction in large-scale applications. Based on data that is available via the open government data initiative, necessary properties and relationships are identified that are necessary to calculate the aforementioned properties.

2.4.1 Sky View factor

Sky view factor (SVF) is one of the primary factors influencing the urban microclimate as it is directly related to the amount of radiative exchange that occurs between the surfaces in the urban environment and the sky. In order to calculate SVF values for our selected study areas in Vienna, our first task was to implement an algorithm in a GIS framework. We choose the CSC method for use with DEMs developed and outlined by Richens (1997).

This algorithm calculates the shadow volume created from a hypothetical light source vector by iteratively offsetting and decreasing the height of a DEM along the vector. This is then repeated for a large number of light sources (e.g. 1000, in this study) spread across the sky hemisphere with a cosine-weighted distribution. The SVF for each pixel is then the fraction of times it is lit to the total number of light sources. In other words, a pixel that is lit by every light source has a SVF of 1 (e.g. an unobstructed view of the sky) and pixel that is always in shadow has a SVF of 0.

In the past this algorithm has been validated both by comparison to hemispherical photography (Lindberg 2007) and simplified geometries with known mathematical solutions (Brown et al. 2001). One limitation of the CSC method is that there will always be significant errors of over estimation at the margins of the result SVF map. This occurs because the input DEM will always have some boundary beyond which there is no information and thus no obstructions of the sky. This can be avoided by simply using DEM maps with a buffer around the area of interest. Using

this technique, it is possible to set the height of the virtual sensor and produce a continuous SVF map at varying heights. For the purposes of validation, the SVF maps were calculated at the height of the fish-eye camera (1m).

Between 12 and 16 photos were taken at each of the 5 study areas with a Nikon Coolpix 8400 with a FC-E9 Fisheye Converter lens mounted on a tripod at the height of 1 meter and pointed directly at the sky. The lens has approximately a 183°field of view. These photos were then processed using the Sky View Factor Calculator version 1.1 created by the Göteborg Urban Climate Group and based on the work of Holmer et al. (2001) and Johnson and Watson (1984).

For each location, we examined a circular area with a radius of 500m from the weather station in accordance with recommendations from Stewart and Oke (2012). We obtained 3 files from the city of Vienna:

1. A DEM including all buildings, trees, and obstructions at a resolution of 0.5m. This kind of file is commonly called a digital surface model or DSM.

2. A DEM including only terrain information at a resolution of 1m or a digital ground model (DGM).

3. A vector file representing each distinct feature of the area as a polygon classified by land use type.

The DSM was used without modification as the model representing the city including the tree canopy. In order to test the efficacy using such a highly detailed DSM in comparison to the more common building only studies, we needed to create a treeless DEM. This was a three step process. First the vector file from the city was used to select all the polygons representing buildings.

Then a 1.5 meter buffer was applied to account for the pixelation of building facades at diagonals to the DEM grid. Finally, the information from the DSM within the buffered vectors was combined with the DGM to produce a DEM containing only buildings. Then each one was used with the CSC algorithm to produce two SVF maps for comparison with

Figure 2.6: SVF algorithm output example.

the hemispherical photographs. One map including tree canopy data and the other with only buildings.

The algorithm produces a grid at the same resolution as the original DSM. The SVF at each grid cell is a value from 0 to 1. When these values are encoded as grayscale values in a raster file format the continuous SVF maps can be viewed as a standard image (Figure 2.6).

We then compared the georeferenced SVF measurements taken in the field to the points sampled in those same locations from the continuous SVF maps. It can be seen that the maps without trees significantly overestimate the measured values (Figure 2.7). It is clear that the continuous SVF maps generated from the DSM including trees correspond better with measured values taken during full leaf season than those generated from a DEM including only buildings.

2.4.2 Aspect Ratio

Aspect ratio is defined as the height-to-width ratio of street canyons (Oke 1981). Due to the topological segmentation of the Vienna base map, an automated calculation of a representative aspect ratio value for an entire street canyon is hardly possible.

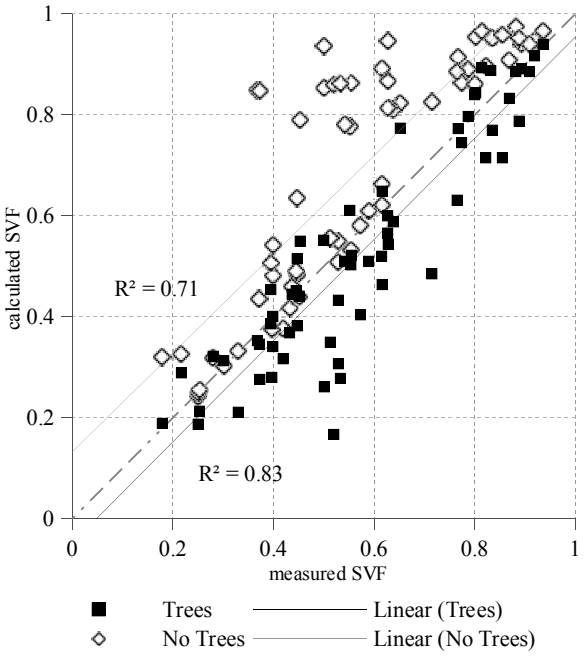

Figure 2.7: Comparison of SVF calculated with and without trees to field measurements.

In Vienna, street canyons are not clearly defined. Streets are not represented as discrete polygons. Unfortunately, the definition consists of various street types (e.g. street lights, walkways, crossroads, road ways, parking spaces, crosswalks, etc.) and therefore typologically diverse entities. Furthermore, there is no standardized procedure on how to handle junctions, differing street widths or differing building heights. In case a consistent theoretical definition can be found that considers semantic data (for instance boundaries derived from the street name) and geometric data (building and street geometry), a feasible solution can be derived even for large-scale data.

The initially developed bottom-up approach applies to street polygons. Street canyons are calculated by identifying street segments that touch a junction or street endings that are defined by the neighboring relations with buildings. Alternatively, the street geometries can be georeferenced with the street names that are administered by the city's traffic department's data that is currently not available via the open data initiative.

The benefit of this approach is that it results in well-defined street

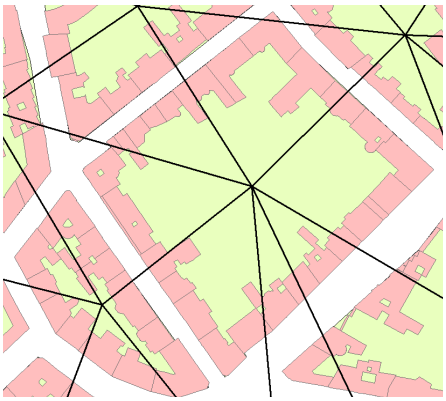

Figure 2.8: Delaunay triangulation based on building block centroids.

canyon geometries. The major drawback is that it requires complex spatial calculations and user input. Furthermore, existing street segments are manipulated (i.e. shortened or separated, combined), leaving the question of what to do with the remaining geometries (i.e. street junctions, traffic circles, etc.) and how to reassign classes to the combined geometries. This procedure includes many iterative steps and spatial operations that increase calculation costs.

To minimize algorithmic costs, the procedure presented in this paper does not focus on an accurate definition of street canyons (both semantic and geometric), but on a superimposed more or less uniformly distributed array of spatial units. Based on the centroids of building blocks, a Delaunay triangulation is made. Based on this triangulation the street facing buildings inside the triangles are identified (Figure 2.8).

The aspect ratio is calculated for each triangle: the mean or median building height is divided by the minimum or average distance. The disadvantage of this solution is that no road type polygons are used and that the calculation focuses on a TIN instead of the real-world street segments. The benefit is the executing performance. Using this approach, the complex calculations are moved towards the end of the algorithm. If the geometry changes for certain areas (i.e. adding or removing of buildings) the aspect ratio calculation is repeated for the affected triangles instead of the whole project area. Algorithm 2.1 describes the logic of the proposed top-bottom procedure.

```
1  Float buildingHeight, canyonWidth;
2  tempBuildingMatrix = list()
3  calculateTIN()
4  for i < getTinList().length do
5      for j < getBuildingBlocks.length do
6          if buildingBlock(j) ∈ getTinList(i) then
7              selectBorderBuildings()
8              updateTempBuildingMatrix()
9              updateBuildingHeight(calculateHeight(mode))
10             updateCanyonWidth(calculateCanyonWidth(mode))
11             removeTriangleFromTinList()
12             setAspectRatio(getTinList(i), getBuildingHeight / getCanyonWidth)
13         end
14     end
15 end
```

Algorithm 2.1: Aspect Ratio.

Figure 2.9: TIN-based results of aspect ratio calculation.

Inside a triangle the street facing buildings are identified. These buildings are stored in a temporary building matrix. Based on this matrix, the respective building heights and canyon widths are calculated and used as input for the aspect ratio calculation at tin level (Figure 2.9). The current implementation does not include inner courtyards into the calculation process.

2.4.3 Built area fraction

The built area fraction calculates the ratio of building plan area to total ground area. The open version of the Vienna base map distinguishes building types into buildings and connections between buildings (for instance, roofed hallways). The accumulated building area is divided by the total ground area.

$$BAF = \frac{A_{unbuilt}}{A_{tot}} \tag{2.2}$$

As it can be seen in Equation 2.2, the resulting built area fraction is used as input to calculate the unbuilt area fraction ($1-builtareafraction$).

2.4.4 Impervious surface fraction

Pervious surfaces are divided into bare soil, vegetated, and water covered areas. The most accurate solution is to divide organic surfaces into classes based on spectral properties. However, this would require spectral information that is not available in the open data catalog. Analyzing the areal photographs and classifying the surfaces is an alternative, but can hardly be automatized as the classification samples have to be assigned manually.

The proposed solution is to divide the existing open data land cover classes due to their assumed spectral properties. Table 2.2 shows the classes (bare soil, vegetated, water) for pervious areas. Sealed surfaces are excluded.

The accumulated area is calculated for each land cover type and then divided by the total ground area. Summarizing the three coefficients (PSF_{soil} for bare soil areas, PSF_{veg} for green areas, PSF_{water} for water covered areas) amount to the pervious surface fraction. The impervious surface fraction is the ratio of sealed horizontal surface areas to total ground area (A_{tot}) without taking buildings ($A_{Buildings}$) into account. Following the previous surface classification process identifies various impervious land cover classes (roads, traffic islands, walkways, pedestrian zones, traffic area on private property, pedestrian areas on

Table 2.2: Land use classes derived from open data land cover properties.

Bare soil	Vegetation	Water bodies
fields, patch, acres, tree farm, agricultural areas	forest, area with tree population	natural waters, swimming lakes
gravel pits	meadows	pools, biotopes on private property
	vineyard	fountains, artificial waters (public property)
	public green area, other unsealed areas	water channels
	sport areas	other water areas
	other vegetated areas	

private property, railways, crosswalks, parking spaces and other traffic areas, building courtyards, walls). The impervious surface fraction is calculated by omitting building types (Equation 2.3).

$$ISF = UAF - (\frac{A_{unbuilt}}{A_{tot}} - (PSF_{veg} + PSF_{soil} + PSF_{water})) \quad (2.3)$$

2.4.5 Mean building compactness

The ratio of built volume (above terrain) to the total building plan area is referred to as the mean building compactness. Building geometries have a relative height property. The relative height is calculated by subtracting the absolute surface height from the absolute building height. The mean building compactness is referred to as the average building height of the project area.

2.4.6 Built surface fraction

The area of building surfaces influences specific physical properties. The used material defines the reflectance, emissivity, thermal conductivity as well as specific heat capacity. By analyzing the surface dimensions, the affected surrounding area can be approximated. To get an idea on how great the amount of built surfaces is in relation to the total built area, the built surface fraction is calculated.

The calculation process defines two modes. As Algorithm alg:bsf shows, mode 1 starts with an extraction of street- and courtyard facing surfaces (walls). All building polygons are split to line features. Every building consists of multiple line segments. Wall segments that are shared by adjacent buildings are removed from the calculation process (Figure 2.10). Analyzing the left and right neighbors of any building polygon for varying building types, identifies these segments. The surface area is calculated by using the wall segment's length and the height of the respective building (Equation 2.4).

$$BSF = \frac{\sum_i l_i \times h_i}{A_{built}} \tag{2.4}$$

Figure 2.10 shows the building line features after the exclusion of shared wall segments. The wall segments are colored due to their respective height. BSF mode 2 follows the approach of mode 1. The difference is that roof areas are also included in the calculation process.

input : A set of buildings of size $b \in buildingBlock$

```
1 for building ∈ buildingBlockh do
2 |   ls=getBuildingLineSegments(building)
3 |   identifySharedLineSegments(ls)
4 |   removeSharedLineSegments(ls)
5 |   calculateBSF(height, length)
6 end
```

Algorithm 2.2: Pseudo code for built surface fraction.

2.4.7 Summary

This section presented an overview of methodological background and recent research efforts that are of use for the following chapters. The building monitoring system MOST and the lessons learned from development and operation build the basis for the work in Chapter 3. MOST is a vendor and platform independent building monitoring toolkit. Data is stored in a topological model (zones and datapoints). This system is deployed in three buildings, and uses a web-based approach and relational data stores. Data access is available via standard industry protocol implementations and web services.

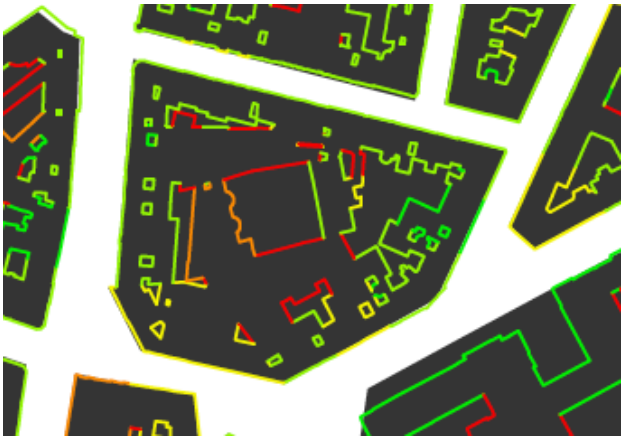

Figure 2.10: Built surface fraction: Wall segments that are shared with adjacent buildings are removed. The height of the wall segments is color-coded.

Modeling the urban microclimate is a complex endeavor and requires a variety of data sources. The proposed algorithms solely focus on static geometry data that is available via open data interfaces. GIS-based analysis of the urban morphology to derive implications about microclimatic conditions in urban settings is a first step towards the GIS-based assessment of urban energy demands that is discussed in Chapter 4.

CHAPTER 3

From Building Monitoring to Urban Monitoring

Building Information Systems handle non-abstract data that is directly related to the built environment (e.g., indoor climate, energy use, building systems' states). Designing a homogenous urban data structure raises the need to consider a variety of data sources and data streams. These might be (i) traffic monitoring data, (ii) weather data, (iii) physical properties of building materials, (iv) occupancy or (v) open government data. As it has been mentioned in the introduction, one characteristic of big data collections is that data might be collected before a specific purpose is defined. As the following chapter shows, a number of performance issues had to be resolved to adapt the proposed monitoring system to a more scalable environment.

As it has been mentioned in Chapter 2, Baldauf et al. (2007) propose a five layered conceptional framework for scalable context-aware systems. These layers are: (i) sensors, (ii) raw data retrieval, (iii) preprocessing routines (iv) storage/management, and (v) application. As this chapter shows, the proposed monitoring systems did not implement such a structure. For instance, data preprocessing routines are directly implemented in the data store. Thus, the execution of preprocessing functionality might affect and even crash the data store. Accordingly, a data

storage concept must consider extensions of the data models as well as adaptions to the overall concept.

As it has been discussed in Chapter 2, data can be structured with a topological model, consisting of zones and datapoints. This work builds upon this concept. On the one hand, zones can be single rooms, or even just spaces inside a room. On the other hand, zones can extend over entire building storeys, even multiple buildings. Sensors can thus represent spatial entities on various scales, reaching from sub-room spaces to entire neighborhoods in the city. The fact, that multi-building zones can not be represented by singular sensors is covered by the extensive use of virtual datapoints. This chapter describes, how the proposed building monitoring system is modified to handle urban-scale monitoring data.

This includes the implementation of a flexible, non-relational datastore, the extraction of processing logic from the datastore to other application layers, and the seamless integration of multi-building and (virtual) sensor communication procedures. Developing an integrative data store that supports multiple buildings proved to be a challenging task, as scaling partitioned relational databases is hard to accomplish and complex to maintain (Leavitt 2010, Hecht and Jablonski 2011).

3.1 Data Store Requirements

The database was initially designed to offer a generic data store for one building. Analyzing the relational database and data processing techniques showed that handling multiple buildings caused severe performance problems. For instance, generating periodic datasets for large time ranges caused performance bottlenecks and eventually crashed the storage service.

This showed that MySQL is not an optimal solution for reading operations in a write intensive environment. Currently, the database handles 80.000 transactional commits, 130.000 inserts, 700.000 selects and 53.000 updates per day with a rapid growing tendency. These are partly caused by preprocessing procedures and partly by sensor communica-

tion. Partitions with varying configurations (e.g. indexes) were introduced to fit different use cases and to distribute the database load (Zach et al. 2012). For example, a partition that handles long-term historical data analysis requires different configurations and resources than an application that needs real time data access. The database performance was assessed by running three testcases that consisted of two scenarios (Table 3.1)

Table 3.1: Database stress test scenarios.

Test scenario	# Datapoints	# Zones	# Values
A	10×10^3	2.5×10^3	2.5×10^6
B	10×10^4	2.5×10^4	$>2.5 \times 10^6$

As the requirement analysis showed, the current database and application design had to be improved in order to support the deployment in an urban environment. Due to the identified problems and bottlenecks in the application design, an alternative data store mechanism had to be developed. The goal was to develop a highly scalable building data storage engine that provides high performance data access. Furthermore, the processing logic should be decoupled from the database logic for three reasons:

- Modifying the processing logic must be possible without modifying the database schema.
- Allowing integration into other projects.
- Resolve database internal performance bottlenecks.

Sensitive and static data, e.g. user profiles and information, and sensor parameter definitions should be separated from the dynamically changing sensor measurements to ease the use of different database configurations.

The relational database schema (Figure 2.4) defines tables that contain seldom-changing data. These are a `datapoint` table that stores sensor metadata (e.g. value ranges, type, sample interval, etc.), a `zone` table that manages datapoints by assigning them to organizational units

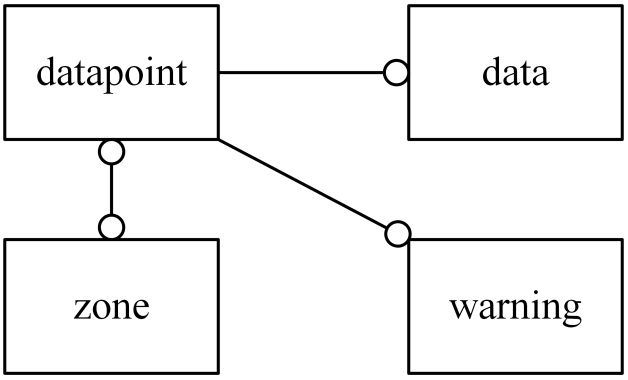

Figure 3.1: High-traffic producing database entities. Excerpt from the ER diagram.

(e.g. a room or building floor) and the `user` table. Sensor measurements are stored in the `data` table. Experience showed that up to 90% of the daily, database-induced network traffic is caused by read and write operations on the `data` table. During the migration process we intended to move dynamic sensor measurements to a NoSQL database and keep the static and security sensitive parts in MySQL. This was established by these steps:

- Seperating the highly dynamic data entities from the configuration data
- Moving data processing routines to a higher application layer to allow parallelization at software level

As demonstrated in Figure 3.1, four objects show a highly dynamic behaviour: `datapoint`, `data`, `zone` and `warning`. Beside the introduced *datapoint, data and zone* tables, the *warning* table also significantly influences the database-induced network traffic. If an error occurs (for instance a broken batch process, or a broken sensor), it is written into the warning table.

To store these data generically, multiple relational and non-relational data stores were analyzed regarding their performance, scalability and data model. Following requirements were elaborated to ensure scalability:

- Ensure transaction security for sensitive master data (for instance configurations, user information)
- Flexible and extensible data model
- High read and write performance for the datapoint value data
- High read performance for the master data store.

Okman et al. (2011) tested NoSQL database security and identified potential security risks. Therefore, the sensitive user data should be stored in a read-only configuration data store and the sensor measurements should be moved to a NoSQL cluster. Transaction security is not a crucial condition for storing sensor measurements. In practice, sensors send a value every minute or even every second. Considering the nature and availability of building sensor measurements (intervals i=[1, 3600] sec), we can conclude that the effect of one lost measurement on an entire day's data collection is negligible.

Due to the reduced number of transactional checks, NoSQL stores provide better access speeds than relational databases. Due to the schemaless data handling, NoSQL databases are predestined for scaling applications. Tudorica and Bucur (2011) compared various databases and found that NoSQL data stores (specifically Cassandra) provide a better write and read latency in a write intensive environment than relational products (MySQL). At approximately 7000 simultaneous read or write operations MySQL became unresponsive.

Building sensor data is not complex. A data tuple consists of a value, a timestamp and a reference sensor ID. Due to this structure a key-value store was chosen for the implementation. Cassandra (Cassandra 2013) provides a powerful query language and is well established in high load environments (BIMserver 2014).

3.2 Database Migration

A Cassandra keyspace can be seen as the non-relational equivalent to a relational database schema. Contrary to relational database schemata,

NoSQL schemata can be changed at runtime. Cassandra organizes data tuples by column families (relational entity equivalent), rows (identified by a row key) and columns (identified by a column key). New tuples are randomly distributed across available nodes (servers). Building data strongly depends on recording time, a sensor location and the observation of a physical phenomenon (the value).

The unstructured NoSQL schema must focus on these three factors (time, source location and value) and the queries that must be supported to offer performant operation. The query structure is a vital point in keyspace design. Due to the lack of a schema, not supported queries have to be implemented in the client application at the cost of performance.

Building performance is quantified by analyzing available data. The proposed framework encapsulates all data access routines in stored procedures. For example, writing operations are provided by the *addData()* procedure. New measurements are checked for constraint violation. If no violation occurs the data will be inserted in the respective table.

To develop a Cassandra keyspace the queries of the stored procedures *addData(), getData(), getDataPeriodic(), getNumberOfValues()* and *delData()* were analyzed. This delivered 23 query logic requirements that must be supported by a non-relational schema. These are, for example:

- Get the latest value of a datapoint.
- Get the latest value of a datapoint starting at a given timestamp.
- Get the number of values for a given datapoint for a given time.
- Get values from datapoint A for a given time range that are higher than the values of datapoint B for the same time range.

A keyspace must support these requirements to be used in a productive environment. As Cassandra distributes data randomly, range queries on data are not natively supported. One approach to solve this issue is to distribute data with an ordered partitioner.

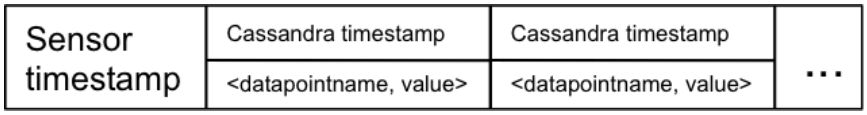

Figure 3.2: First Cassandra keyspace design.

NoSQL databases as Cassandra offer this possibility at the expense of access performance. Due to the performance shortcomings, this practice was not considered in the design phase. The first keyspace design draft started with a simple analogy (Figure 3.2). A sensor timestamp specifies the row key, as it is unlikely that two measurements are recorded at the same UNIX-timestamp. As a key-value schema can be changed at runtime, new data tuples (consisting of a Cassandra generated timestamp column key, the datapoint id and a value) are added horizontally or vertically. Relational databases do not allow this practice at runtime. Adding new columns requires a new schema definition.

Unfortunately, this first draft already violated a query definition. It was not possible to get the latest measurement of a sensor. As Cassandra distributes the data tuples randomly, it is not possible to search through data tuples without accessing all tuples in the database. The schema definition was continuously updated and tested to fit the requirements and to optimally use the Cassandra resources.

Every keyspace prototype was analyzed for scaling performance. Data recording intervals usually range between minutes and hours. To develop a sustainable scaling strategy, calculations assume that a sensor stores values every second. This assumption means that one sensor executes about 3.15×10^7 operations per year. According to Equation (3.1), the size for one tuple T is calculated by the column key size and the value size.

$$T = v + k + t + t_0 \tag{3.1}$$

Values v, keys k and timestamps t are stored as 64-bit IEEE-754 floating point. Cassandra produces a 15 bytes overhead per tuple (t_0). This includes an additional 8 bytes overhead for counter and expiring columns.

All in all, this results in 39 bytes per tuple, including overhead. One sensor that stores sensor ID, timestamp and value produces about 1.23×10^9 bytes = 1,18 GB data per year when a new measurement is sent to the database every second. Cassandra allows a maximum size of 2 GB per column and 5 TB per file. This means that one sensor can store 4237 years of data in one file, without the need to scale.

This approach was refined, as searching through 4237 years of randomly distributed data is not an optimal solution. The database was partitioned by creating monthly shards. New data tuples are added as columns to a monthly row for each sensor. This improves querying, as the access is restricted to one row per sensor for one month's data. For a 30-day cycle this results in 2.6×10^6 measurements with a size of 96 MB for each row.

Besides working on the NoSQL keyspace, the configuration database was optimized for reading operations. Processing logic was extracted from the stored procedures and moved to a Java module. The initial MySQL schema (Figure 2.4) defined stored procedures to read, write and process data. The SQL queries that are encapsulated in the stored procedures were divided into three groups:

- Compare values or datasets.
- Generate periodic data.
- Validate for constraint violations.

Constraint violations are: (i) deadband, (ii) allowed maximum and minimum values, (iii) sample interval, and (iv) necessary minimum sample interval. Depending on the constraint definition, a violation results in either discarding the current value or generating a warning.

Querying data requires data comparison. Time or value ranges are compared by boolean operators. Based on certain modes that define which method is used, periodic datasets are calculated. The SQL queries were translated to Java and moved to a new module, the `most-preproc` library (Figure 3.5). This allows an inclusion in other projects and simplifies code maintenance.

3.3 Application Architecture

A scalable architecture is realized by a distributed system that is based on loosely coupled modules for

- data processing
- data retrieval
- data persistence
- data access
- data presentation

To reduce the database load that is induced by processing routines the responsible logic was extracted from the database and moved to an indepedent module in a higher application layer. To realize modularity, scalability and to increase reliability on the application level, modules can be deployed redundantly and on different machines.

Stateless core components allow new instances to be added during runtime, for instance to serve load peaks (i.e. monitoring occupancy during the morning rush hour) and to be removed in the cooling-down period.

Such a concept implementation requires a central distribution mechanism that routes requests between modules, respective physical machines that are distributed via a city's buildings. As the proposed framework is written in Java, all components are bundled by a Message Oriented Middleware (MOM) that is accessed via a Java Message Service (JMS) API.

The communication process is established by dynamically created queues (point-to-point) and topics (publish-subscribe). On binary protocol level, the Advanced Message Queuing Protocol (AMQP) was chosen, as it is determined as secure, reliable, high performant and vendor-neutral (Vinoski 2006).

Every single instance of redundantly deployed components listens to the same queue. The transferred request messages are always handled by exactly one instance (the one who initially took the message from

the queue). Changes in datapoint values can be observed by subscribing to a datapoint's topic. That specific topic is identified via the owning datapoint's unique name and the prefix *OBSRV_*: *OBSRV_*<*dp-name*>. For instance, the activity observation of the specific sensor `con1` (contact sensor) would be realized via the topic id `OBSRV_con1`.

3.3.1 Virtual Datapoints and Virtual Datapoint Collections

Datapoints always represent physical entities (for instance sensors). As has already been mentioned, the concept of virtual datapoints (VDP) realizes data representations that are not bound to a physical sensor. For instance, this might be the on demand calculation of an entire building's accumulated energy consumption, average temperature or requesting values from a weather forecast. Furthermore, virtual datapoints can be used to describe physical phenomena that extend over multiple buildings (for instance, accumulated energy consumption in an entire neighborhood).

As VDPs work at a very low application level and behave like native datapoints, other application parts can use them transparently. VDPs can be seen as plug-in components and are extensively used to support data simulation, model calibration and data prediction.

The implementation of a virtual datapoint is deployed as a distinctive component and is accessed through a dynamically created queue which's name is derived from the VDP's type by prefixing *DATA_*: *DATA_*<*type*>. For instance, a VDP that provides Radiance Radiance 2014 simulations could use the queue `DATA_RADIANCE`. VDPs that implement the same type share the same queue, which realizes support for multiple instances and hence increases reliability and availability.

VDPs offer a way to establish communication between buildings and components within one application context, which is an essential step towards a distributed urban monitoring and simulation structure. However, VDPs can only return a primitive value, for instance the aforementioned overall energy consumption. To decrease configuration effort for buildings we introduce Virtual Datapoint Collections (VDC) that combine a set of VDPs to a bundle. This bundle is configured once and applies standard-

ized calculations on the respective building's data. A VDC uses the same communication process and protocols as described in the VDP concept.

3.4 Results

3.4.1 Database Migration

The final keyspace design defines a column family for every datapoint in the database (e.g. `datapoint1`, Figure 3.3). Query access is restricted to datapoint entity level. If data is randomly distributed, Cassandra only allows queries on keys, as keys are indexed. Sensor data are stored in a timely linear matter. No sensor can monitor phenomena that take place in the past or the future, just at a certain point in time. Therefore, data within a datapoint column family is organized by 12 rows. These rows are monthly shards with the first day of a month as a row-key. This practice further restricts the sorting problem to a monthly level. If a sensor stores a value every second, a search algorithm must access one month of data which equals a maximum of 2.6×10^6 randomly distributed values.

Every data tuple consists of a *timeUUID* that is secondary indexed and the measured value. *timeUUID* columns are sorted by the time component first and then by the raw bytes. This ensures that the measurement timestamps are unique. By applying a secondary index on the unique measurement timestamp, the need for sorting in query procedures is removed, as a secondary index provides sorted values, although a random partitioner is used. Cassandra defines a maximum of 2 billion cells (rows $\times$ columns) by partition.

Considering the proposed example (a sensor records a measurement every second), one sensor executes 3.15×10^7 operations per year. To optimize data access the data is divided into 12 monthly shards. This means that one partition can hold 1.6×10^8 measurements, which equals 5.3 years of data before new partitions are necessary. If sensors commit a measurement every minute, one partition can hold approximately 300 years of data per sensor. The maximum file size of 5TB is never reached, before the maximum number of cells is exceeded.

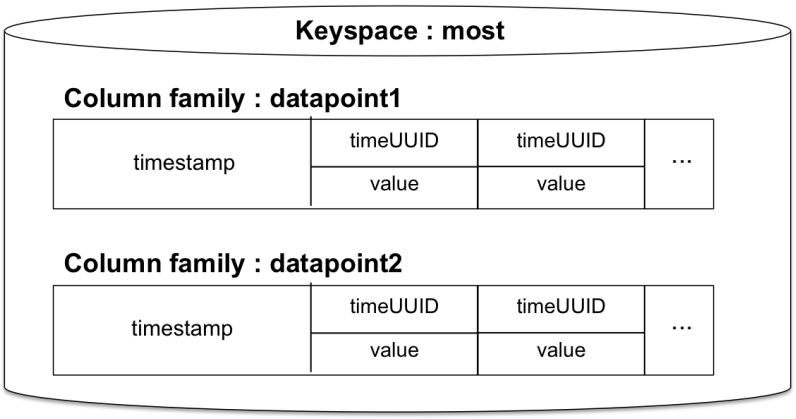

Figure 3.3: Final range query supporting Cassandra keyspace design.

3.4.2 Data Access

The proposed framework provides an API to access building data via various services. As Figure 3.4 shows, a persistence layer provides access to sensor measurements and configuration data and handles the communication of relational (MySQL) and non-relational (Cassandra) data store.

The building information system's static configuration (sensor specific parameters, user infor-mation, zonal information) is moved to a MySQL database that is optimized for read operations, as the configuration data are infrequently changed. The storage engine was changed from InnoDB to MyISAM and the key buffer was increased. The `ROW_FORMAT` was fixed, and more RAM installed. A feature that updates configuration data by accessing a building information model from a BiM-server instance (BIMserver 2014) is being developed.

The `most-data` module handles the available data storage implementations. The building information system core can connect to three data storage engines - a MySQL database, a Cassandra or neo4j NoSQL cluster. By using the provided API, support for additional storage technologies can be implemented without adjusting the core modules. Periodic data generation algorithms and data validators are moved to a separate `most-preproc` library that is included in the respective database module implementations (Figure 3.5).

Standardized dataset definitions are used to exchange data between modules and libraries. All datapoint datasets are exchanged between

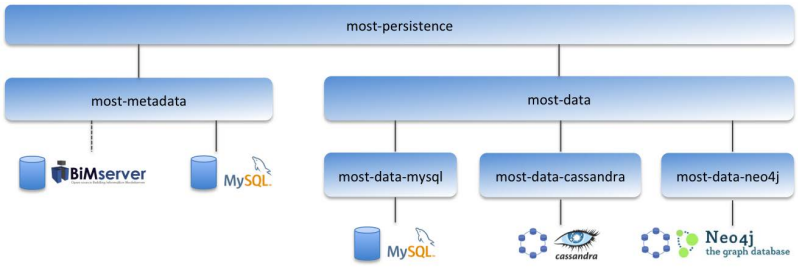

Figure 3.4: Persistence layer structure with three supported data stores.

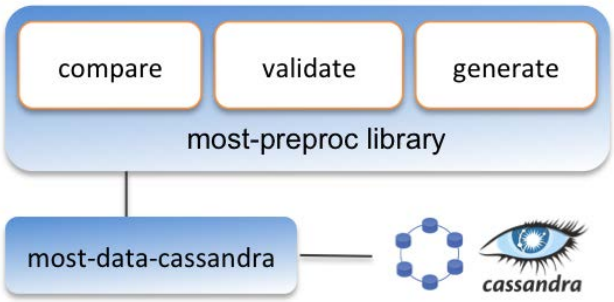

Figure 3.5: Database - preprocessing libary communication.

modules by one instance, the datapoint entity manager.

3.4.3 Application Architecture

The application is divided into four layers. As Figure 3.6 shows, these are a persistance, service and presentation layer, as well as a service adapter that offers access to the data collection via standardized protocol implementations (OPC Unified Architecture, oBIX, RESTful). The presentation layer consists of two client applications, a web client and a mobile client. The service layer implements the BMS business logic (data processing) and the VDCs. The persistence layer manages the relational master data store and the NoSQL datapoint value store. To allow a distributed module deployment, all components communicate via the MOM.

Figure 3.7 illustrates the specific module implementations of the persistence layer after the revising process. Configuration and security sensitive data is stored in a MySQL database. Sensor configurations are partly kept in BIM models that are managed by a BiMserver instance. The metadata-module handles configuration requests and accesses the two databases. Datapoint values are either stored in the well tested initial

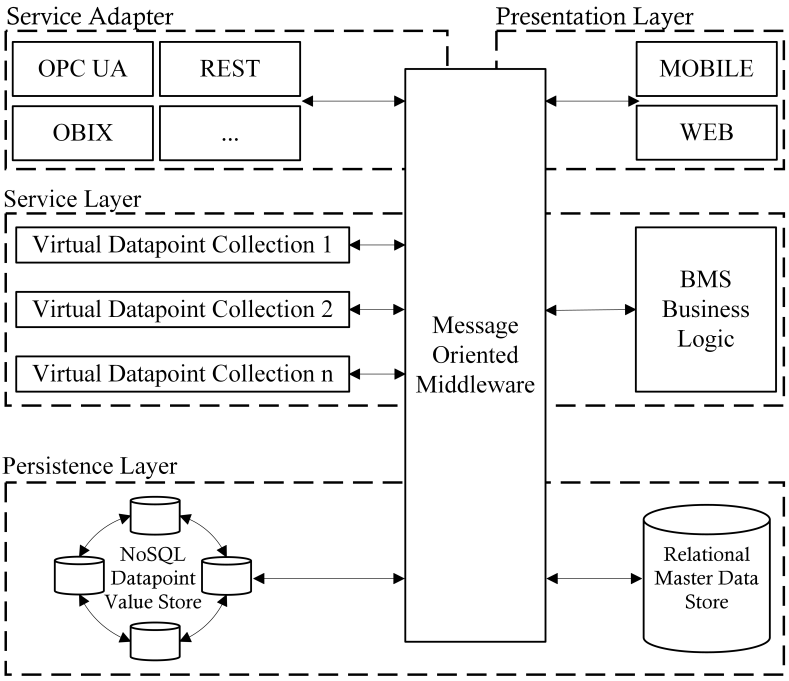

Figure 3.6: The proposed urban monitoring framework's system architecture.

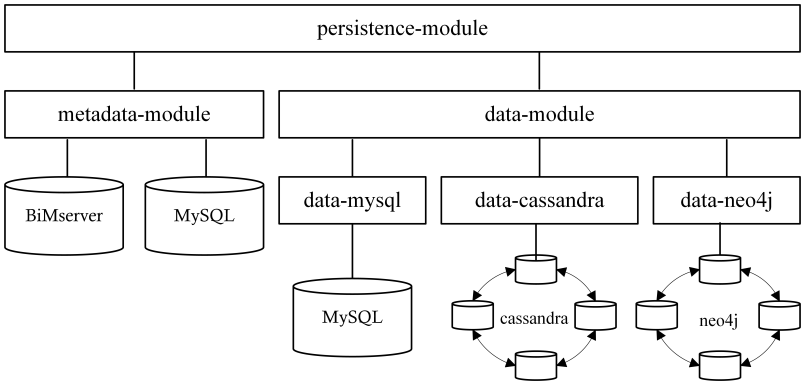

Figure 3.7: Persistence layer: specific module implementation.

MySQL environment, in a Cassandra cluster, or a neo4j graph database.

Neo4j support was introduced partly to monitor relationships between urban entities and partly to store sensor measurements. The configuration data and datapoint data is merged and connected in the persistence-module. This module also implements the processing functionality that was extracted during the refactoring process.

3.5 Conclusion

This chapter presented an approach to implement a scalable building monitoring system. The proposed methods intend to prepare the existing infrastructure to offer its functionality to other applications. Besides our efforts that mainly focus on large-scale building monitoring, other, closely related research efforts focus on the small-scale building and user monitoring. For instance, Yao et al. (2015a) developed a prototypical implementation of a smart home by deploying sensors to the environment and to instrument people as sensors, by attaching them with RFID tags. It is intended to equip the proposed monitoring system with more human related data sources. Future work will utilize people and their mobile devices as sensors to communicate subjective impressions about the building environment. Within the web of things domain, a number of research efforts exist that offer synergies with the proposed system. For instance, Jung et al. (2014) develop an integration middleware for the internet of things. IoTSyS implements various connectors to proprietary standards (KNX 2016, M-Bus 2016) and offers technology specific oBIX (oBIX 2016) object implementations. Qanbari et al. (2015) developed a middleware called *Gatica* that enriches real-time sensor data using annotations to support the direct analysis of IoT sensor data on resource-constraint IoT devices (e.g. gateways). All this efforts at our research institution show that affected disciplines focus on developing scalable, generic approaches instead of relying on monolithic, proprietary solutions.

3.6 Summary

This chapter describes how the proposed building monitoring core modules were refactored to support data streams from multiple buildings simultaneously. The datastore was analyzed, and dynamic sensor measurements were moved to a NoSQL cluster. Configuration data is kept in a relational datastore that is optimized for read operations.

Beside the implementation of a non-relational datastore that supports multiple buildings, processing logic has been moved from the database

to a processing library. Data communication was generalized by moving communication between entities to a message oriented middleware.

CHAPTER 4

GIS-based Energy Demand Assessment

The following chapter describes efforts to generate geographic building models (gBIM) from two dimensional GIS-data and to develop a mathematical framework to utilize these models to develop GIS-based energy demand assessment methods. Three-dimensional, topological building models are generated based on the two-dimensional GIS-input data that are available via open data interfaces. The mathematical framework of the ISO 13790:2008 standard is simplified to support gBIMs. The manually calculated, standard compliant energy demands of a sample of buildings are compared to the results of the automated version introduced in this chapter.

These simplified energy demand assessments are calculated according to the ISO 13790:2008 *Energy performance of buildings - Calculation of energy use for space heating and cooling* (ISO 2012) standard. The framework for calculating energy certificates is defined in the European Union Directive 2010/31/EU (EU 2010) and is well established on national level ISO-Standard implementations. The national implementation of this standard for Austria, which is published as ÖNORM B8110-6 (ASI 2014) and ÖNORM B8110-5 (ASI 2011) is used to create a lightweight mathematical model for the calculation of GIS-based energy certificates. Due to the two-dimensional nature of the available GIS-data and the lack of

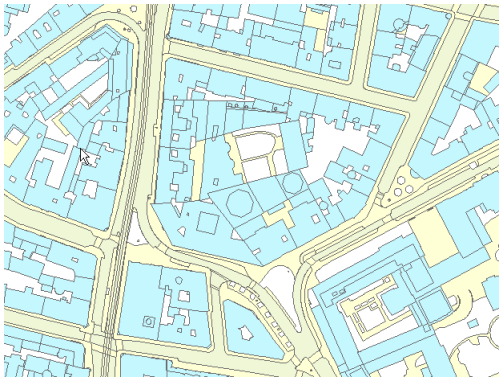

Figure 4.1: Schematic map of the study area.

certain attributes (e.g. wall materials, retrofit documentation, roof geometries, etc.), the mathematical model of the standard is simplified to handle GIS-based input data.

For this purpose, a solid data basis must be established that (i) is able to store and provide building data for multiple buildings and (ii) automatically generate all the necessary semantic data to allow the usage of the gBIM models to calculate energy demands. Based on Open Goverment Data a procedure is introduced that derives all the necessary geometric and attributive information needed to run basic energy demand calculations.

4.1 Input Data Preprocessing and Building Model

Open government data is published under a Creative Commons Licence CC BY 3.0 AT - (Commons 2014) that allows to manipulate, transform, and redistribute the data in any form. The procedure discussed in this paper can thus be applied to any city that offers open data interfaces. Vienna implements a variety of web services to access this data in various proprietary and open source raster and vector formats (e.g. DXF, TIF, JPEG, SHP, CSV, XML, JSON). The procedure was tested in a study area in the inner city (high-density urban location). As it can be seen in Figure 4.1, the study area is characterised by diverse building morphologies, well-defined building blocks, irregular street networks, and varying land uses.

The geometric base data is offered via the Viennese cadastre. Every building has a unique ID. Before the actual processing starts, attributive data is attached to the corresponding geometry. This includes

- building area
- building perimeter
- relative building height
- height above digital elevation model (DTM)
- number of storeys
- construction period.

To implement a feasible demand and supply calculation framework, a building model must be created that offers an applicable level of detail whilst guaranteeing query and processing speed. The building model consists of three entities: `Building`, `Storey`, and `Element` (Figure 4.2). `LineSegments` always depend on a parent `Element` and solely describe the geometry.

Buildings consist of various storeys that represent either a ground floor that shares an external wall with soil, a regular storey or a storey that has a shared wall with an attic/roof. Depending on this storey type and the construction period, basic physical properties, such as heat transmission coefficients are determined.

Each storey has multiple elements. An analysis of the storeys and the surrounding/neighboring buildings allows to determine the orientation of the element, the type (shared with other buildings, e.g. fire wall or exposed to outdoor climate), and the shading (every element is part of a storey and inherits the relative height of the storey).

Assigning elements to multiple storeys violates topological consistency. An element should only belong to a single storey instance, but the element's geometry can be shared with other buildings. Another design guideline is that adjustments to physical element properties should be distinct from operations on the geometry. Furthermore, an element

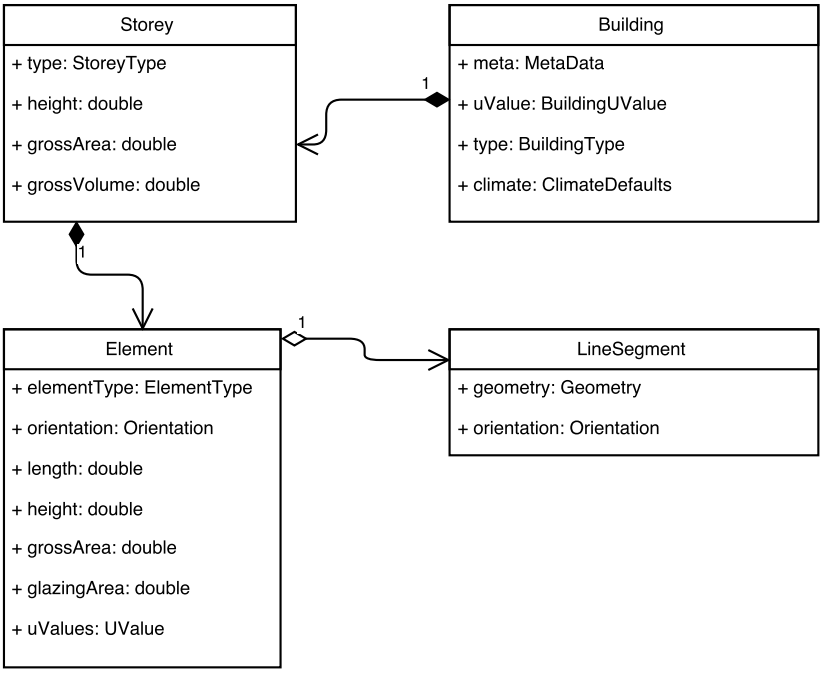

Figure 4.2: gBIM model with shared geometries

can belong to only one storey, but the element's geometry (one or more line segments) can belong to multiple elements. Storeys and elements inherit certain properties such as unique ID, type reference etc. from the parent building they belong to.

As it can be seen in Algorithm 4.1, the USIM building model is created in four steps. Every new building needs a geometry representing the building footprint. The method `mapBuildingGeometry()` has to determine the type of geometry (does the building geometry contain holes - e.g. inner courtyards, ventilation shafts) and to test for topological correctness. The sample application utilizes the JTS Topology Suite, but any collection of GIS-algorithms is feasible. This step assures that the building outline polygon can be used to derive correctly oriented building wall elements. If the topological correctness of the building footprint geometry is assured, the semantic building properties are calculated and/or imported (gross/net area, perimeter, building id, relative building height, number of building storeys, height above see level, building type and year

of construction). Depending on the location and the height above sea level, the default climate settings are determined. If no custom climate data (e.g. weather file) are used during the import, the default settings are used. These define the average outdoor temperature for the building's location and the heating days per month according to the standard. After the building and all of it's properties are successfully mapped the details of the building model (storeys and elements) are generated based on the geometry and base attributes (Algorithm 4.1). After the determination of a topologically correct two-dimensional building footprint, the building is now brought to the third dimension.

input : A set of buildings B
output: gBIM of B

1 **foreach** *building* $i \in B$ **do**
2 Building b = new Building();
3 mapBuildingGeometry(B_i);
4 mapSemanticBuildingProperties(B_i);
5 determineClimateSettings(B_i);
6 createBuildingModel(b);
7 **end**

Algorithm 4.1: Create gBIM building entities and map building properties.

With the creation of the basic building entity from GIS data, a black box containing basic geometrical, physical, and metereological properties can now be used to derive building storeys. As it can be seen in Algorithm 4.2, the storeys of the building are determined by the total count of storeys. The average storey height $h_S\ [m]$ is the fraction of the building height relative to the sea $h_{B_{rel}}$ level and the total count of storeys of the respective building c_S (Equation 4.1).

$$h_S = \frac{h_{B_{rel}}}{c_S} \tag{4.1}$$

As the correct building height is very important for the heating demand calculations, it is vital to have accurate heights. In the proposed case, relative building heights $h_{B_{rel}}\ [m]$ are calculated by intersecting Digital Surface Models h_{DSM} with Digital Terrain Models h_{DTM} (Equation 4.2).

$$h_{B_{rel}} = h_{DSM} - h_{DTM} \tag{4.2}$$

Within the USIM building model, there exist three types of storeys that can be assigned to the respective entities (*GROUND*, *ROOF*, *DEFAULT*). These types are used to specify default physical properties of the buildings that influence transmission and ventilation losses and influence the respective storey's elements (eg. shading). As Algorithm 4.2 shows, there is no specific type for attics and roofs. As there is hardly any information regarding roof geometry available, the first gBIM version only supports roofs as a type of storey. A building storey belongs to exactly one building entity and inherits certain properties such as the geometry, climate settings, and dimensional information (e.g. average storey height) from the parent building (`mapBuildingPropertiesToStorey()`).

input : An imported Building b.
output: gBIM constructional model.

```
1  for i < b.getStoreyCount() do
2      Storey s = new Storey();
3      if i == 0 then
4          s.setType(StoreyType.GROUND)
5      end
6      else if i == b.getStoreyCount() − 1 then
7          s.setType(StoreyType.ROOF)
8      end
9      else
10         State s.setType(StoreType.DEFAULT)
11     end
12     mapBuildingPropertiesToStorey(s);
13     createElementsForStorey(s);
14 end
```

Algorithm 4.2: Create gBIM storey entities.

Based on the building storeys, wall elements are generated (Figure 4.3). Building storeys consist of n elements, which represent distinct wall surfaces ($element \in storey \in building$). This means that all elements in a building inherit from the base building's geometry and attributes. The energy demand calculation is mainly depending on the wall elements and their corresponding attributes (Figure 4.2):

- **Element type**

Determines whether the element is exposed or not (e.g. shared fire wall).

- **Orientation**

  The orientation is used to determine the influence of solar radiation on this particular wall element.

- **U-values**

  Based on the building's construction period, default U-values are assigned that influence the thermal performance of the element.

- length
- height
- gross area
- gross volume.

input : A Building storey s
output: Wall elements $\in$ s

```
1  Element e;
2  Geometry geom = s.getGeometry();
3  for LineSegmentls : geom.getLineSegments() do
4      e = new Element();
5      e.setGeometry(ls);
6      e.setProperties(s);
7      if sElementExposed(e) then
8          e.setType(ElementType.EXPOSED)
9          calcDefaultGlazingArea(e);
10     end
11     else
12         e.setType(ElementType.COVERED)
13     end
14     setDefaultPhysicalProperties(e);
15     calcOrientation(e);
16     calcAbsAngle(e);
17     calcShadingFactor(e);
18     s.addElement(e);
19 end
```

Algorithm 4.3: Determine gBIM element type based on geometry and building properties.

Elements are created by splitting the storey's footprint vectors into line segments. A line segment consists of two pairs of coordinates that form a

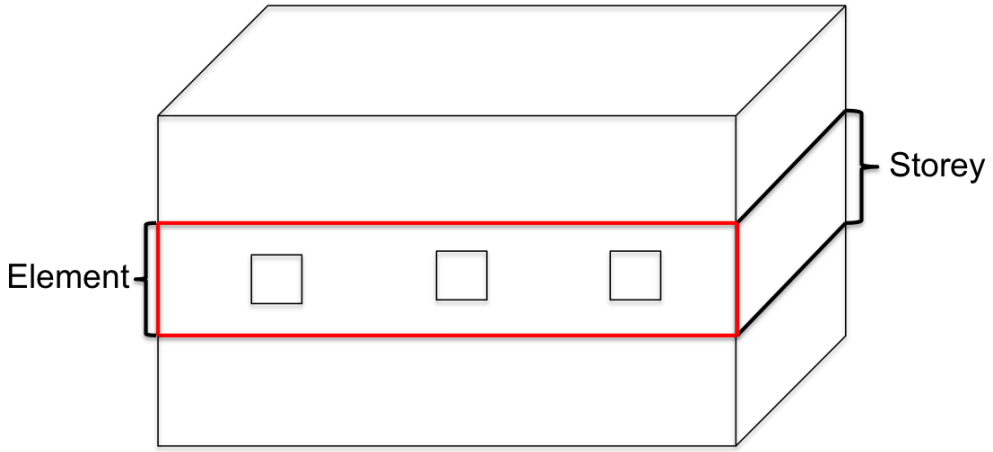

Figure 4.3: Basic topological building model (*Building*, *Storey*, *Element*).

straight line with a unique orientation and depending on the storey height shading and glazing factor. As it can be seen in Algorithm 4.3, currently an element can have only one line segment. The element's height is specified by the average storey height h_S. The element gross area a_g $[m^2]$ (Equation 4.3) is calculated by multiplying the characteristic base length of the element l_E $[m]$ with h_S $[m]$.

$$a_g = l_E \times h_S \tag{4.3}$$

Beginning at ground level the first storey is analyzed. Following the building footprint the wall elements are derived from the input data. Beside the glazing factor, area, characteristic length, height, and shading factor, each element is classified by an element type. An element can be shared with other buildings and thus not be directly exposed to the outdoor climate. Exposed elements belong to walls that are either entirely exposed (e.g. facing a road, courtyard, etc.) or to storeys that are higher up than the roofs of adjacent buildings (Figure 4.4). After the element type is determined, the orientation that determines the angle of the element towards north and the shading factor are calculated (Algorithm 4.3).

The building geometry is available in the MGI Austria GK M34 system (EPSG:31256), which is measured in meters to the east and north of the central meridian. This projection is designed to support large

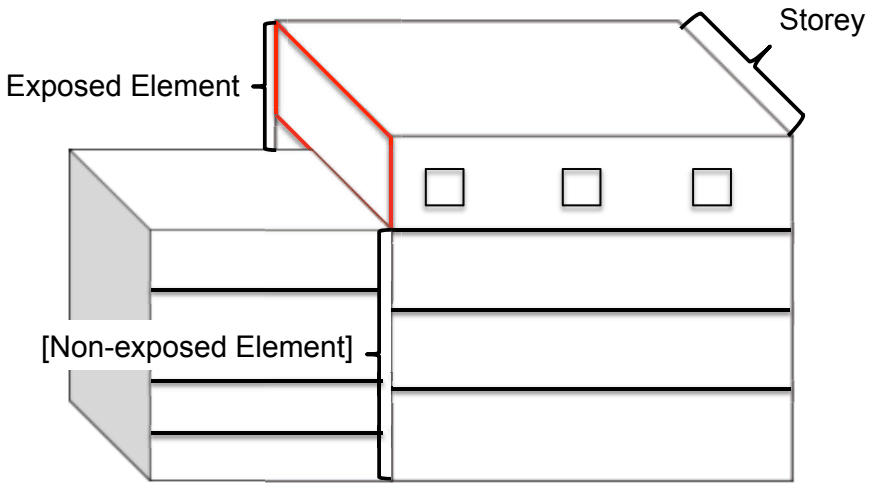

Figure 4.4: Element type determination.

and medium scale topographic mapping and engineering surveys (BEV 2006). The orientation of wall elements and the corresponding glazing area is equivalent to the cardinal direction. Utilizing the fact that the coordinates are available as north and east pairs allows the determination of the orientation with minor computation efforts. As it can be seen in Figure 4.5, beginning at an arbitrary starting point, a local cartesian coordinate system is applied. Based on this, the angle ϕ between north and the vector that defines the line segment is calculated using Equation 4.4.

$$\cos\phi = \frac{\vec{n} \times \vec{e}}{\|\vec{n}\| \times \|\vec{e}\|} \tag{4.4}$$

The generated gBIM models are then used to calculate GIS-based energy certificates. To calculate the energy demand, the Austrian implementation of the ISO 13790:2008 standard that is defined in the ÖNORM B 8110-6 and ÖNORM B 8110-5 was chosen. These standards are used to calculate energy certificates on a yearly and monthly basis in Austria. The following sections describe the necessary steps to calculate monthly energy estimates (heating, cooling demand, internal gains, solar gains) based on the generated gBIMs. The goal is to develop a lightweight method that allows to simulate energy demand and supply at an urban level. The annual heating demand per reference area $Qh\ [kWh * a^{-1}]$is calculated as described in Equation 4.5.

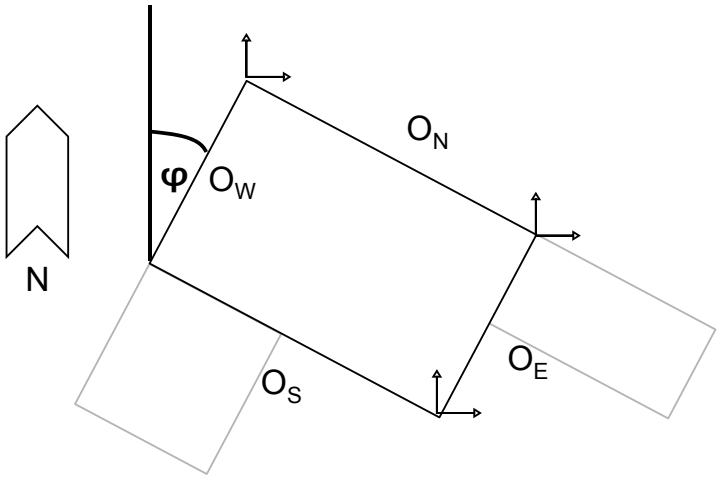

Figure 4.5: Orientation of buildings. Local cartesian coordinate systems are applied to calculate the orientation angles.

$$Q_h = Q_l - \eta \times (Q_i + Q_s) \tag{4.5}$$

Q_h consists of the building's heat losses Q_l, the internal gains (Q_i), and solar gains (Q_s).

4.2 Heat Losses (Q_l)

Heat losses Q_l $[kWh/m]$ (Equation 4.6) are specified by heat losses due to transmission Q_T (Equation 4.7) and ventilation Q_V (Equation 4.8).

$$Q_l = Q_T + Q_V \tag{4.6}$$

$$Q_T = \frac{1}{1000} \times L_T \times (\theta_i - \theta e) \times t \tag{4.7}$$

$$Q_V = \frac{1}{1000} \times L_V \times (\theta_i - \theta e) \times t \tag{4.8}$$

In Austria Q_T is based on a minimum indoor temperature of 20°C and Q_V is based on a maximum indoor temperature of 26°C. This means that Q_T refers to the amount of energy that must be supplied to a building to guarantee the minimum indoor temperature of 20°C during heating periods and to guarantee the maximum indoor temperature of 26°C during cooling periods. Assuming a very basic case (heat losses per

month, standard values for heating and cooling, no special HVAC, and non-complex geometries) the monthly heat losses due to transmission (Q_T) and due to ventilation (Q_V) strongly depend on transmission (L_T) and ventilation (L_V).

L_T $[W/K]$ describes the transmission conductance of an entire building and is calculated as

$$L_T = L_e + L_u + L_g + L_\psi + L_\chi \tag{4.9}$$

$L_\psi + L_\chi$ represent additional terms addressing two- and three-dimensional thermal bridges. L_u is the thermal conductance for building parts that connect the conditioned internal space with the outdoor air via unconditioned spaces (e.g. winter gardens). As it is hardly possible to derive this information from 2.5 dimensional GIS data, L_ψ, L_χ, and L_u are neglected. This simplification leads to Equation 4.10.

$$L_T = L_e + L_g \tag{4.10}$$

The thermal conductance L_e $[W/K]$ is defined in Equation (4.11).

$$L_e = \sum A_k \times U_k \tag{4.11}$$

It consists of the thermal conductance of all building parts that connect the conditioned internal building space with the unconditioned surrounding space (e.g. external walls). In the proposed building model, L_e is calculated for all elements that have the asigned type *EXPOSED*. A_k is calculated by accumulating the area of all k building elements with a one-dimensional thermal conduction with the respective U-Value (U_k) for the respective building parts. For buildings that do not offer a sophisticated data base, standard values that depend on the construction period are assumed (Table 4.1). As it can be seen in Table 4.1, doors and roof areas are not considered. U-values are assigned to the respective elements during the import process of the GIS data. As it can be seen in Algorithm 4.4, the first storey in a building (ground floor) and the uppermost storey only consist of one element. This means that currently only flat roofs are assumed. Furthermore, every element has the default

U-values for external walls and windows assigned. In the current implementation, windows are not defined as distinct elements, but derived from the wall elements.

input : An array of storeys.
output: Storeys with thermal default properties.

```
1  for i = 0; i < storey.length; i++ do
2      for i = 0; i < storey.length; i++ do
3          if si.REFERENCE == 0 then
4              e.setKD(default.kd)
5          end
6          else if si.REFERENCE == storey.length − 1 then
7              e.setOD(default.od)
8          end
9          e.setAW(default.aw)
10         e.setFE(default.fe)
11     end
12 end
```

Algorithm 4.4: Mapping U-values to building elements.

Table 4.1: Default U-Values per epoch: CT= Construction Epoch, KD = Basement Ceiling, OD=Uppermost Storey Roof, AW=External Wall, FE=Windows (OiB 2015).

CT	KD	OD	AW	FE
$<$ 1900	1.25	0.75	1.55	2.5
$<$ 1945	1.20	1.20	1.50	2.5
$<$ 1960	1.10	1.35	1.30	2.5
$>$ 1960	1.35	0.65	1.20	3.0

Beside L_e, also L_g is included in the proposed model. L_g refers to the thermal conductance for building parts that connect conditioned internal space to the ground (e.g. basement, cellars).

As it can be seen in Equation 4.12, Ventilation term (L_V) is defined by multiplying the volume dependent heat storage capacity of air $c_{p,L} \times \rho_L$ $[\frac{Wh}{m^3 \times K}]$ with the air change volume ν_V $[\frac{m^3}{h}]$, with $c_{p,L} \times \rho_L = 0.34$ (ASI 2014).

$$L_V = c_{p,L} \times \rho_L \times \nu_V \tag{4.12}$$

Assuming a base case without mechanical and only natural ventilation, Equation 4.13 can be used to calculate the air change volume.

Depending on the building type and location, the air change rate $n_{L,FL}$ $[h^{-1}]$ can range between 0.3 to 0.5 h^{-1}. This means that the entire air volume is exchanged in between 20 and 30 minutes.

$$\nu_V = n_{L,FL} \times V_V \tag{4.13}$$

The net air volume V_V $[m^3]$ is calculated by multiplying the net ground area $0.8 \times BGF$ with the average storey height h_s (Equation 4.14).

$$V_V = 0.8 \times BGF \times h_s \tag{4.14}$$

4.3 Internal Gains (Q_i)

Internal gains combine thermal energy produced by electrical devices, artificial lighting and the occupants. On a monthly basis, Equation 4.15 can be used to calculate Q_i $[kWh/m]$.

$$Q_i = \frac{1}{1000} \times q_{i,h,n} \times BGF \times 0.8 \times t \tag{4.15}$$

Currently, two building types are implemented (residential and office buildings). Depending on these types the average net internal gains $q_{i,h,n}$ $[W/m^2]$ are used (Table 4.2). Based on these values, the net ground area $BGF \times 0.8$ and the usage time per month t $[h/M]$ are used to calculate the internal gains.

Table 4.2: Default net internal gains $[W/m^2]$ for heating/cooling and implemented building types (ASI 2011).

	Residential	**Office**
Heating	3.75	3.75
Cooling	-	7.5

4.4 Solar Gains (Q_s)

Beside the building specific losses and internal gains, solar gains depend on the exposure of the building to solar radiation. The different impact of solar radiation on the varying building materials (concrete, transparent) must be considered to reach appropriate results. As it has already been

mentioned, it is not possible to derive exact window positions from GIS data. Nevertheless, it is possible to derive the exact amount of exposed wall elements. Depending on the derived elements, a general glazing area of 15% of the respective element's net area $a_{e_{net}}$ $[m^2]$ is assumed to be window area (Equation 4.16).

$$a_{G_e} = a_{e_{net}} \times 0.15 \tag{4.16}$$

With an element depending window area a_{G_e} it is now possible to derive the transparent/ glass area (Equation 4.17).

$$A_g = f_g \times A_W \tag{4.17}$$

The shading factor F_S determines the reduction of solar radiation regarding the shading of the building due to (i) shading through other buildings, (ii) shading through surface texture, (iii) building parts, (iv) location of the window. Previous research showed that there are promising ways to specify shading properties, either by using the sky view factor (ratio of the visible sky from a certain point on a street or facade) or aspect ratio (height to width ratio of street canyons). Implementations for both approaches are documented in Chapter 2. Both approaches are furthermore documented by Glawischnig et al. (2014a). The current approach aims to increase the processing speed.

A detailed calculation of F_S requires information on the shading due to topography F_h, overhangs F_o and lateral overhangs F_t (Equation 4.18).

$$F_S = F_h \times F_o \times F_t \tag{4.18}$$

It is assumed that F_o = F_t = 0, due to the fact that two-dimensional data cannot be used to identify vertical overhangs. Based on the surrounding topographical model, the simplified shading factor is specified as (Equation 4.19):

$$F_S = F_h \tag{4.19}$$

The shading factor is applied due to surrounding neighbours and increases from a building's bottom to top storey and has a value between

0 and 1. This input data is generated by applying a buffer around buildings (currently 500m) and considering the height of neighboring buildings. The resulting solar gains $Q_S\ [kWh/M]$ (Equation 4.20) are a product of global solar radiation on a surface with a certain orientation per month ($I_{S,j}\ [kWh/M]$ (ASI 2011) and the actual collector area of transparent surfaces *k* with the orientation *j* ($A_{trans,h,k,j}\ [m^2]$).

$$Q_S = \sum_j (I_{S,j} \times \sum_k A_{trans,h,k,j}) \tag{4.20}$$

Q_S is calculated on a monthly basis. The average amount of global solar radiation I_S is defined as a second degree polynomial (Equation 4.21) with a_2, a_1, a_0 being sea level dependent monthly coefficients derived between 1971 and 2000 (ASI 2014). $h\ [m]$ describes the building's height above sea level. The height above sea level refers to the digital terrain model at the building's geographic location (also see Equation 4.2).

$$I_S = a_2 \times h^2 + a_1 \times h + a_0 \tag{4.21}$$

$A_{trans,h,k,j}$ (Equation 4.22) is determined by the exposed elements' specific glazing area, the shading factor and the windows' U-value (Table 4.1).

$$A_{trans,h,k,j} = A_g \times F_S \times g_w \tag{4.22}$$

4.5 Results

We now present the results of the proposed procedure when applied to an urban setting in Vienna. The GIS-energy certificates are compared to manually calculated standard energy certificates according to the Directive 2010/31/EU, to identify differences between the automated, large-scale and manual method. The introduced method is implemented in a use case implementation that covers parts of the inner district in Vienna with approximately 700 imported buildings (Glawischnig et al. 2014c).

As it can be seen in Figure 4.8 the test area consists of a large variety of building morphologies. All the buildings in the sample area were

processed by the method introduced in this paper. From the stock of imported buildings, twenty were randomly selected to test the quality of the results. For these twenty buildings, detailed energy certificates were calculated and were compared to the results of the automated large-scale calculation.

Overall, the results of the heating demand calculation showed an average deviation of 15% and a median deviation of 13% between the respective Energy Certificate and the GIS-based method. For buildings with an overall consistent outline, the automated method showed surprisingly accurate results. The best performing building only showed a deviation of 2 kWh between the proposed method and manually calculated transmission heat losses (Table 4.3).

Figure 4.9 shows the interface of the sample implementation. The building is located in Canovagasse, in the first city district. As it can be seen in Table 4.4, the best results that were achieved showed a difference in Q_T of 2 kWh, whilst the most error prone buildings showed a difference of 112 kWh. The maximum error in the sample is 42%, the best $<$ 1% (Figure 4.6).

The first test run showed that buildings with a homogenous, not too complex footprint display very promising results. Buildings with complex geometries do not perform very well due to certain properties that are automatically derived during the import process. For instance, Figure 4.7 shows the building footprint of the worst performing building. The building has a very complex footprint and as a more detailed analysis showed, less glazing area than was assumed in the importing process (narrow vertical walls without windows).

Furthermore, the roof structure is included in the energy certificate whilst the proposed method and data only assumes flat roofs. Thus, the tower is not included at all. These anomalies are yet to be included in the importing process.

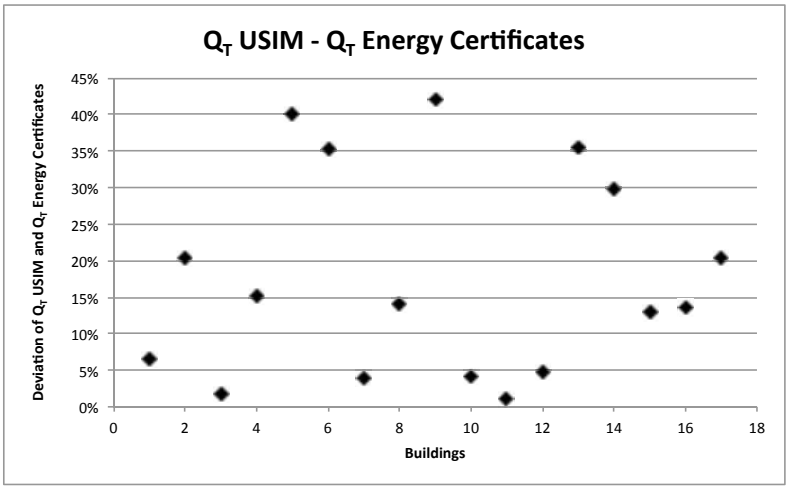

Figure 4.6: Results of the first test run. Comparison of gBIM Q_T to manually calculated Q_T.

Figure 4.7: Worst performing sample building with complex geometries and roof structure.

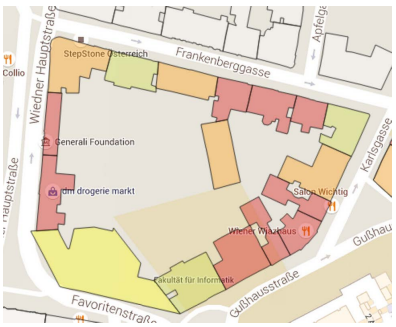

Figure 4.8: Sample building block with calculated energy demand.

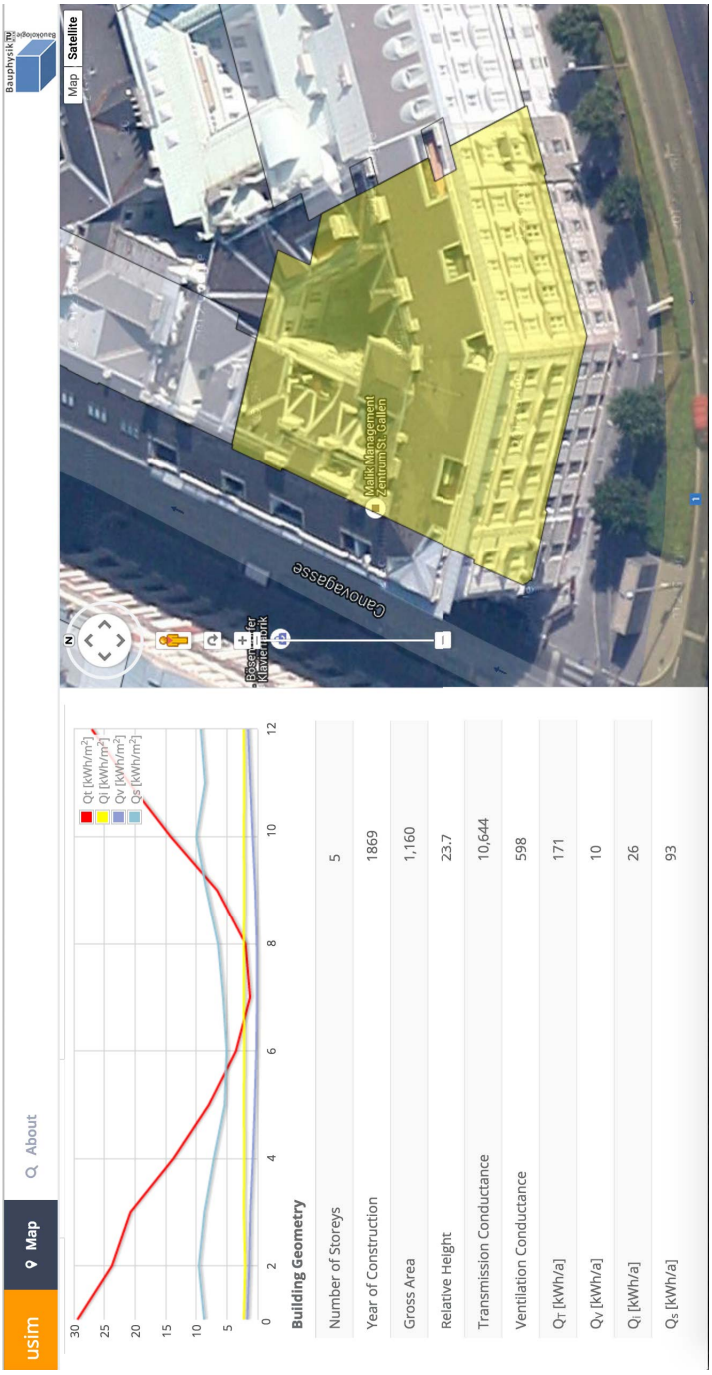

Figure 4.9: Sample building with calculated energy demand.

Table 4.3: Comparison of L_T and Q_T between Energy Certificates and GIS-based method.

Factor	USIM	Energy Certificate
L_T	10644	10405
Q_T	171	173

Table 4.4: Comparison of Q_T between Energy Certificates and GIS-based method for the entire sample.

USIM Q_T	Energy Certificate Q_T
234	250
190	239
214	217
165	140
106	177
117	181
217	226
262	225
266	154
258	269
171	173
96	101
200	129
239	168
291	253
240	207
285	227
188	175
275	259

4.6 Summary

This chapter presented an approach to derive low-detail geographic building models (gBIM) from open government GIS data to calculate simplified energy certificates according to current legal regulations. Two-dimensional GIS-data is analyzed and transformed to three dimensional building models. The procedures to derive necessary input (geometry, attributive data) are described as well as the necessary simplifications of the mathematical framework that needed to be applied to handle not-supported input. The Austrian implementation of the ISO 13790:2008 *Energy performance of buildings - Calculation of energy use for space heating and cooling* (ISO 2012) standard, which is published as ÖNORM B8110-6 (ASI 2014) and ÖNORM B8110-5 (ASI 2011) is simplified to

handle gBIM and the results of the automated method are compared to manually calculated energy certificates for a set of 20 buildings.

CHAPTER 5

Urban Retrofit Scenarios

The gBIM models and the respective prognosed energy demands offer a starting point for the analysis of urban retrofit scenarios. The analysis of large-scale energetic optimization potential is a valuable tool to reach energy and climate relevant sustainability goals on an urban level. Urban planning considers a variety of input factors, such as economic, social and environmental influences. These influences, for instance global warming potential, are not mapped in the current version of the retrofit model. The calculation of retrofit costs solely depend on material and labor costs, and thermal conductivity.

The retrofit module that is introduced in the following sections investigates the optimization and modernisation potential of buildings, and entire neighborhoods based on physical parameters. The potential energy demands that are calculated using the GIS-based energy assessment approach are used as a basis to calculate retrofit scenarios.

The following chapter describes (i) the method that is used to calculate costs and payback period of retrofit scenarios, (ii) how the user input influences the simulation and cost calculation, and (iii) the building product ontology that is used to manage the building product instances and the respective properties (labor costs, material costs, physical properties, etc.).

5.1 Retrofit Scenarios

As it can be seen in Figure 5.1, the definition of retrofit scenarios starts with the gBIM base scenario that originates from the initial import process. This scenario comprises a building instance with the building geometries, and additional properties, such as the standard monthly and annual heat transmission and ventilation losses, as well as solar and internal gains.

Whether the building instance uses a calibrated energy use assessment method or the solely geometry-based approach introduced in this paper does not affect the retrofit computation mechanism. The proposed retrofit module depends on various parameters that can either be provided by the core module or must be provided manually via user input. The boundary conditions are used to start the calculation process in the core module. As it is explained in Figure 5.1, building product data is obtained from a building ontology. Currently, the ontology supports a relatively small number of classes, instances, and corresponding properties, but can easily be extended.

Table 5.1 shows that manual interaction is only necessary to specify the desired U-value, which should be achieved within the scenario. The retrofit area that has to be considered within the scenario is automatically calculated by the number of storeys of the respective building and the corresponding number of exposed wall elements.

The thickness of the retrofit material, which has to be applied in order to reach the desired U-Value improvement, is a function of U-Value difference and thermal conductance of the insulation material. The thermal conductance and other insulation and plaster material properties, as well as material costs and labor costs are provided by the building product ontology.

The assessment of retrofit scenarios starts with the determination of user-specified boundary conditions. Currently, the proposed module supports the setting of the wall and window U-values to influence the retrofit calculation. The estimated original gBIM U-value $U_{original}$ and the user-

Table 5.1: Minimum required properties for retrofit scenarios with data source and unit.

Parameter	Source	Unit
Retrofit Area	gBIM	m^2
ΔU	User	W/(m^2K)
Thickness	gBIM	m
Insulation	Product Ontology	
Plaster	Product Ontology	
Material Cost	Product Ontology	EUR
Labor Cost	Product Ontology	EUR

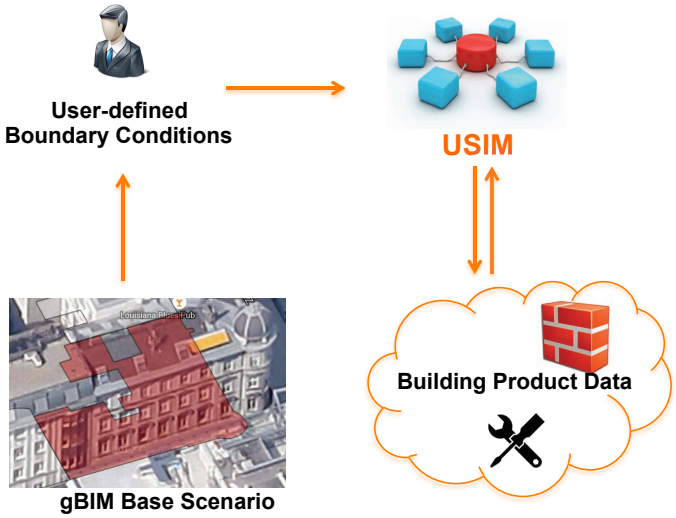

Figure 5.1: Main parts of the USIM retrofit workflow.

specified deviations $U_{desired}$ are used to calculate the desired U-value difference ΔU (Equation 5.1) that is used for the computation of the respective scenario.

$$\Delta U = \frac{1}{U_{original}} - \frac{1}{U_{desired}} \tag{5.1}$$

Based on the available material properties (thermal conductance) that are entailed in the product ontology, the necessary thickness of the respective insulation material is calculated with Equation 5.2. ΔU is the U-value improvement that has to be achieved in the current scenario. $\lambda_{product}\ [w/mK]$ describes the insulating capacity of the respective product.

$$d = \Delta U \times \lambda_{product} \tag{5.2}$$

5.2 Ontology

As it has already been mentioned, an ontology about building products is used to determine the costs and realizability of the retrofit scenarios. An ontology describes a formal specification of a certain domain, thus makes the domain of interest formal and machine manipulable. According to Gruber (1993, pp. 1), an ontology is "... an explicit specification of a conceptualization".

Ontologies offer an efficient way to model the building domain regarding (i) building element, and (ii) building product dependencies. The goal of the product ontology is to define the building retrofit domain so that a reusable set of building related terms is available. This enables an automated search for building products in the WWW, for instance via web crawlers.

The Web Ontology Language (OWL) was chosen to create the building product ontology. OWL builds upon the Resource Description Framework (RDF) and as such semantically extends RDF. The proposed product ontology is designed to answer questions regarding material compatibility. These could for instance be:

```
What insulation products can be used in combination with a
specific mortar to guarantee a certain U-Value?
```

Building product data is obtained and stored in a building product ontology. The application of an ontology allows a seamless, automated data updating process. Data entities can easily be added and removed without affecting the initial building model. The ontology builds upon an abstract building that consists of certain products (Figure 5.3). These products fit certain purposes (e.g. brickwork mortar, insulation, etc.). All the building products instances have properties, such as *labor price, material price, thermal conductivity, etc.*

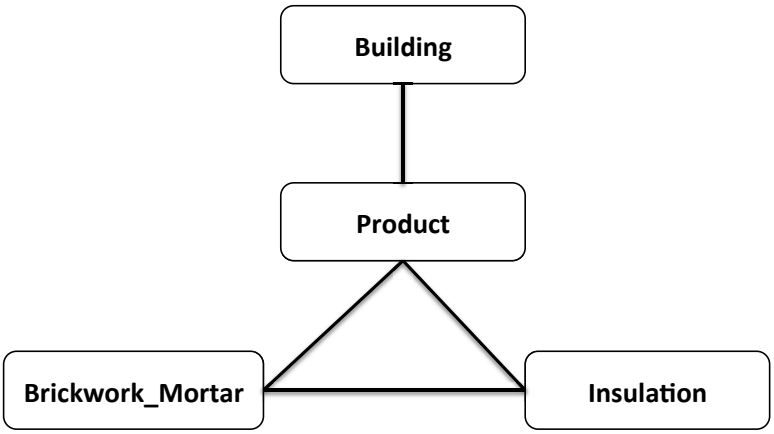

Figure 5.2: Product ontology classes and relationships.

5.2.1 Classes

The proposed product ontology starts with a `Building` class. The building contains `Products` that currently consist of `Brickwork_Mortar` and `Insulation` materials. Figure 5.2 gives an overview of all the named classes that are currently available. To answer the domain specifying question, a product class is needed that shares relationships with mortar and insulation materials, as shown in Figure 5.2.

As argued above, the current version of the ontology does not contain a class for material producers. This is due to the circumstance that only default instances that represent an average set of properties (e.g. material costs, labor costs, thermal conductivity) for a certain product class are provided.

5.2.2 Properties

To define the domain of interest that can be seen in Figure 5.2 and Figure 5.3 it is necessary to provide certain object properties that establish a relation between the classes. Table 5.2 gives an overview of the available object properties. If an insulation material is compatible with a certain mortar, it can be assumed that the mortar is also compatible with the insulation material. Thus, `supportsInsulationMaterial` is the inverse property of `supportsMortar`.

Beside the object properties, the data properties that can be seen

Table 5.2: Product ontology object properties.

Level 1	Level 2
productSpecific	supportsInsulationMaterial
	supportsMortar
	compatibleWith
	compatibleWithForUValue
pricePerUnit	hasLaborCost
	hasMaterialCost
ensuresThermalConductivity	

in Table 5.3 are supported. Currently, the labor costs and the material costs can be set. The thermal conductivity is also mandatory to calculate the amount of insulation material that is needed for the retrofit scenario. As there are only manually added default instances available, product producers and the respective properties were omitted.

Table 5.3: Product ontology data properties.

Level 1	Level 2
pricePerUnit	labor
	material
thermalConductivity	
unit	

5.2.3 Instances

Within the product ontology, material costs and labor costs are already accumulated per defined unit, based on Baubuch (2013) or Dam et al. (2014). For instance, as Figure 5.3 shows, the class building has a subclass *EPS_Insulation_Panel* (EPS_Insulation_Panel $\in$ Synthetic $\in$ Insulation $\in$ Product $\in$ Building).

Based on the class definitions, the final ontology is populated and tested with a number of individuals. A default individual is manually created for each mortar and insulation group. Material and labor costs, as well as physical properties are average values that were calculated by considering a number of representative products from various producers. The current implementation contains a small number of entities, but can be easily extended. For instance, the properties of the default EPS-insulation panel instance can be seen in Table 5.4.

Table 5.4: Default ontology instance for EPS-insulation panels

Property	Value	Unit
Thermal Conductivity	0.035	w/mK
Material Cost p.U.	1.13	EUR
Labor Cost p.U.	2.0	EUR
Unit	1	m^{-3}

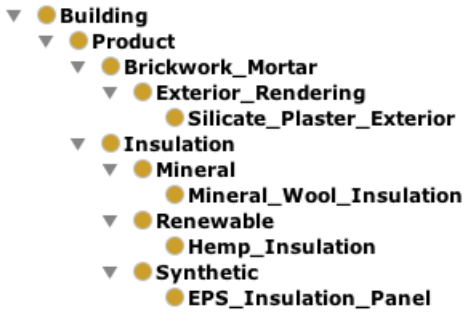

Figure 5.3: USIM building ontology.

With the proposed ontology, the domain question

`What insulation products can be used in combination with a specific mortar to guarantee a certain U-Value?`

can now be answered with the following query:

```
Mineral_Wool_Insulation that supportsMortar some
(Silicate_Plaster_Exterior that thermalConductivity value
0.035)
```

5.3 Cost Calculation

The material and labor cost information from the building product ontology are used to calculate the accumulated costs of the retrofit scenario. As shown in Equation 5.3 the material costs for the defined entire building retrofit scenario, are calculated by accumulating the material costs for every storey $C_{material}$. The exposed area of the respective storey $A_{exposed_i}$ is multiplied with the necessary thickness of the insulation material d, and

the standardized material price p. Labor costs are calculated accordingly, with one difference (Equation 5.4). An optional weighting factor w_i can be applied to cover increasing costs for varying building heights that result from scaffolding and security measures.

$$C_{material} = \sum_{t=1}^{n} A_{exposed_i} \times d \times p \tag{5.3}$$

$$C_{labor} = \sum_{t=1}^{n} A_{exposed_i} \times d \times p \times w_i \tag{5.4}$$

5.4 Workflow

The retrofit scenario depends on a user defined input regarding a desired optimization of the walls' or windows' U-values ΔU (Equation 5.1). As it can be seen in Figure 5.4, the user provides an update to the simulation parameters (results to a new ΔU). Furthermore, the user choses the retrofit materials that should be used in the current scenario.

The `SimulationController` creates a simulation object that contains the necessary information and triggers the actual simulation computation in the `SimulationService`.

The `SimulationService` retrieves the material and monetary properties from the building product ontology and calculates the current retrofit scenario. The thickness of the material, the retrofit costs, and the amortisation duration are calculated and returned to the client.

The `BuildingService` always holds an instance of the active building that is available as long as the controller exists. All services are singletons. This way, the active building can be used by new plugins within the entire context, as long as the user has an active session.

5.5 Amortisation

As the selected building is always available via a service, it is possible to access it from anywhere in the web-application context. This fact was used to add an amortisation plugin to the retrofit module. Based on the active building and the current retrofit scenario the estimated total costs

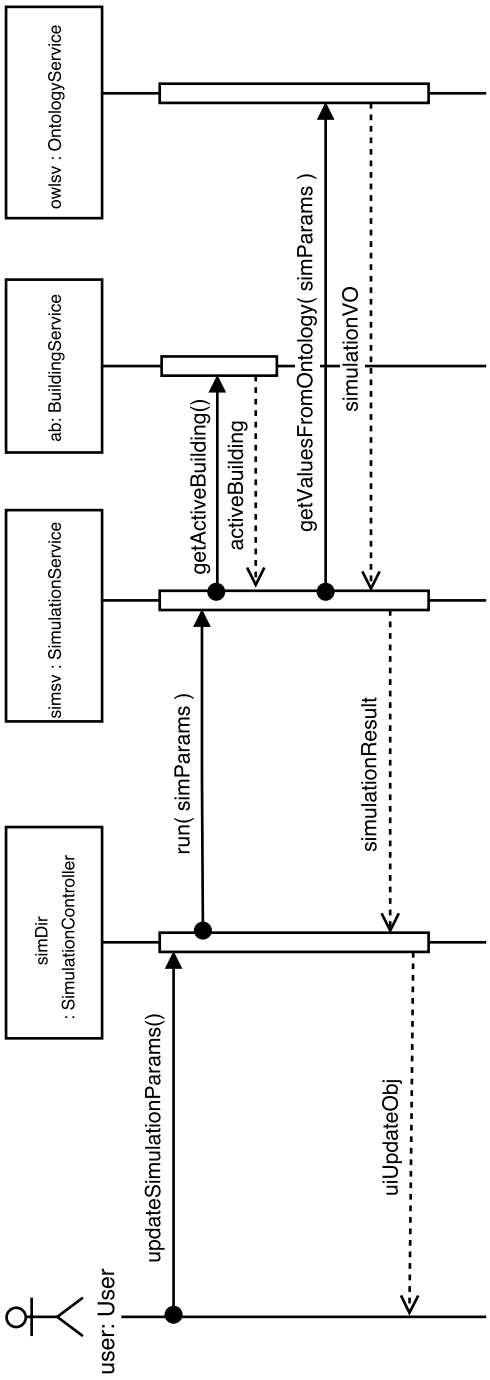

Figure 5.4: Simulation Sequence Diagram.

are amortised if the condition defined in Equation 5.5 is satisfied. G_t refers to the energy savings per month that would be achieved when realizing the current retrofit scenario.

$$\sum_{t=1}^{n} G_t > A_0 \tag{5.5}$$

As it can be seen in Equation 5.6, the savings per month are calculated by multiplying the difference of the energy demands Q_T from the base scenario Q_{T1} and the active retrofit scenario Q_{T0} with the specified price for electrical energy c (in EUR/kWh). The energy price is a factor that can be manually changed by the user.

$$S_y = \sum_{t=1}^{n=12} (G_{T1_t} - G_{T0_t}) \times c \tag{5.6}$$

Beside the variability of the energy price, the user is able to specify an interest rate. In this case, S_y is extended to Equation 5.7 by multiplying the monthly savings with the interest rate r $[EUR]$. The monthly savings are calculated in relation to the interest rate and accumulated on a yearly basis to get the yearly savings S_y.

$$S_y = \sum_{t=1}^{n=12} ((G_{T1_t} - G_{T0_t}) \times c) - ((G_{T1_t} - G_{T0_t}) \times c \times r) \tag{5.7}$$

Based on the logic and data that is exposed via the respective services, the web-application and it's modules can easily be extended.

5.6 Summary

Introducing a building product ontology allows an automated extension and update cycle without human interaction. This chapter showed how a simple ontology can be used to answer building product domain specific questions regarding the compatibility of insulation materials for retrofit scenarios. A well formulated ontology can support a dynamic, rapidly growing retrofit application. The proposed ontology only contains a small amount of materials, properties and instances, but can be quickly extended. Beside the ontology, the simulation sequence steps that have to

be executed to calculate a new retrofit scenario, are introduced. These steps range from the user's retrofit specifications to the calculation of the scenario with the utilization of the proposed building product ontology.

CHAPTER 6

Prototypical Implementation

A real-life implementation and application scenario is tested at the example of the inner city of Vienna. As it can be seen in Figure 6.1 the building monitoring system MOST is used to provide the building monitoring data. As MOST offers a number of data exchange mechanisms, it is possible to integrate it without major implementation efforts. To demonstrate the generic nature of the data model and data handling procedures that were introduced in Chapter 3, aside from the monitoring data that is communicated via the MOST context, a weather station is directly communicating the measurements to the USIM core.

Beside the monitoring data, building geometries are imported via the Viennese open data interfaces, which support a variety of standardized geospatial web services for data exchange. These standards specifications for Web Map Services and Web Feature Services were developed by the Open Geospatial Consortium and are documented by Vretanos (2005).

The building monitoring data, sensor data, and the open government data are communicated via RESTful web-services to the core module. There, the monitoring data is processed, as it is described in Chapter 3. A web application is provided to allow human interaction with the toolkit. Building product data is provided by the product ontology.

This chapter introduces the web-application that is used to implement client-side access to the varying USIM-modules (building statistics, mon-

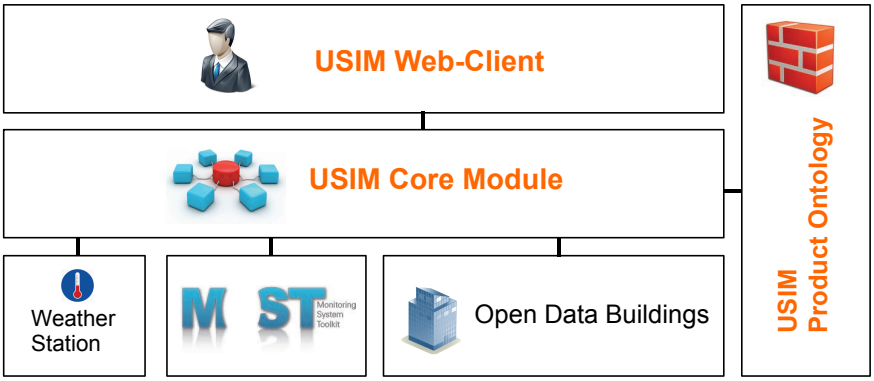

Figure 6.1: Prototypical Implementation Module Overview.

itoring, energy demand assessment, and retrofit scenarios). The various modules expose the respective core functionality and allow users to interact with the methods that were presented so far.

6.1 Monitoring Module

To demonstrate the generic data store mechanisms, two data sources with varying data structures are included. The MOST system provides building data in a topological structure (zones and datapoints). As the USIM data store uses a similar concept, data integration is established without compatibility problems.

As it can be seen in Figure 6.2, zones and datapoints are seamlessly integrated. The test zone (Department of Building Physics and Building Ecology) entails a number of different sensors. The active sensor is a temperature sensor. This zone and all of its datapoints are managed by MOST. USIM solely establishes a data connection and reads measurements periodically. This means that changes to the monitoring logic of the integrated monitoring systems are immediately taking effect in the proposed application.

The sensor measurements are processed by MOST and then forwarded to the proposed core module via RESTful web services. Buildings can be added and removed on the fly, as long as a standardized data access is possible and provided by the respective building's monitoring system. To demonstrate this concept, not only a building, but also

a distinct data producing entity with its own monitoring and data handling context is connected to USIM. Figure 6.2 shows that building instances and sensors can be separately addressed.

As the thermometer icon on the map indicates, the sensor is a weather station that sends its data to the core module. Beside the continuous integration of the external MOST data, the weather station only communicates with the system core, if new measurements are recorded. The DAVIS Vantage Pro2 (Davis 2015) is mounted at the roof of our department building and operates via the standard DAVIS console (Figure 6.3). Whenever a new measurement arrives, it is forwarded to USIM and processed accordingly in the core module.

6.2 Energy Demand Assessment Module

To test the GIS-assessment methods of energy demands, approximately 900 buildings are processed. The project area that can be seen in Figure 6.4, is located in the inner districts of Vienna, near the Technical University and has an extend of about 3 km^2. The project area represents the densely built city center morphology.

Typical Viennese settings entail well-defined building blocks, diverse buildings, irregular street networks and varying land uses. The buildings in this area are classified by building age. The exact number of buildings per construction period can be found in Table 6.1.

Table 6.1: Total building count by construction period of the project area.

Construction Period	Building Count
$<$ 1900	761
1900 - 1945	28
1945 - 1960	119
1960 - 1975	10
1976 - 1993	7
not specified	7

The building geometries are provided by the core module. The implementation of the energy demand assessment method is entirely client based. Moving the calculation work to the client side removes a lot of stress from the servers. Furthermore, changes to the assessment library

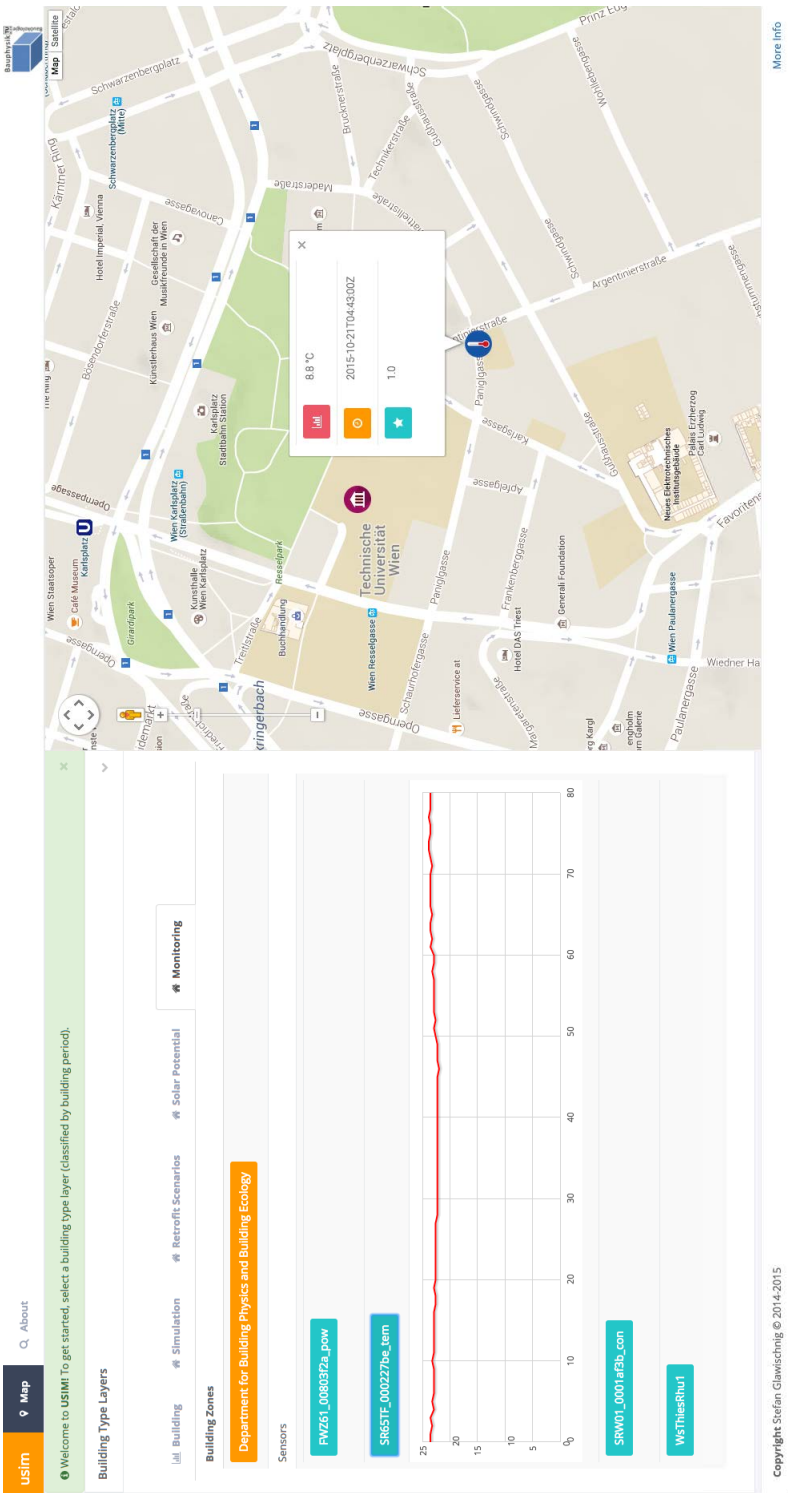

Figure 6.2: Demonstration of Monitoring Module.

Figure 6.3: DAVIS Vantage Pro2 mounted at the TU Wien.

Figure 6.4: Selected project area.

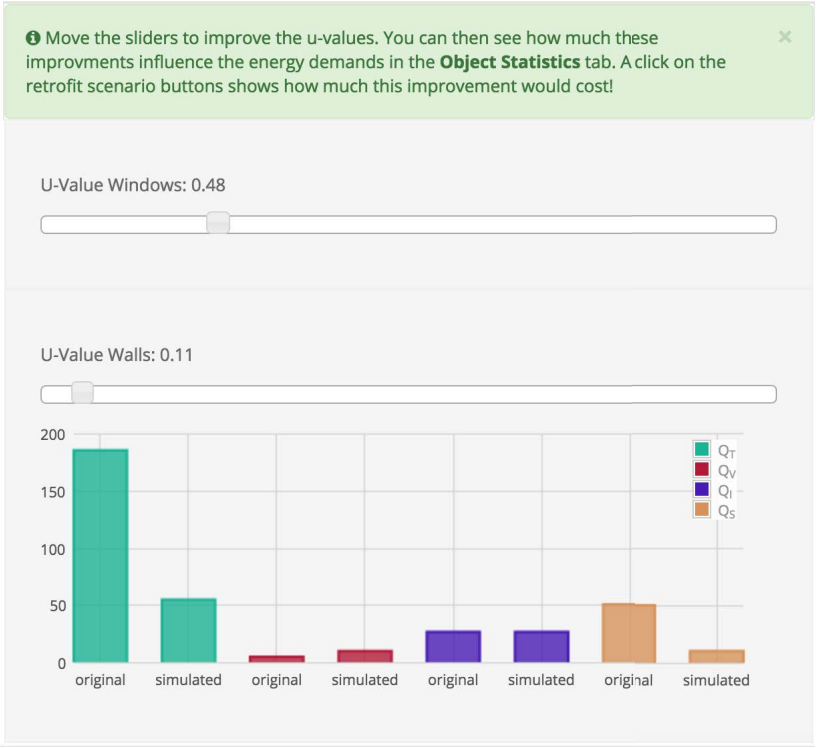

Figure 6.5: Visualization of how different U-value settings influence the heat losses due to transmission.

can be made without raising the need to redeploy the servers. As demonstrated in Figure 6.5, energy assessment depends on manually applied U-Value settings. The yearly improvements that can be realized by the given adjustments are immediately visualized in a bar chart, which shows the accumulated values for a year. Due to the non-complex nature of the mathematical framework, changes of the U-Values for windows and walls can be immediately processed and included in the energy balance equation of the respective building. The effects are included in an updated calculation of transmission and ventilation losses and solar gains.

The assessment library can be enhanced easily. Currently, the energy demand is calculated on a monthly basis. As Figure 6.6 demonstrates, the updated U-Values are immediately applied to the energy balance equation of the respective building. The improvement is calculated for every month and visualized in a graph. To achieve the calculated improvements, the Retrofit Module offers the tools to calculate retrofit options and costs.

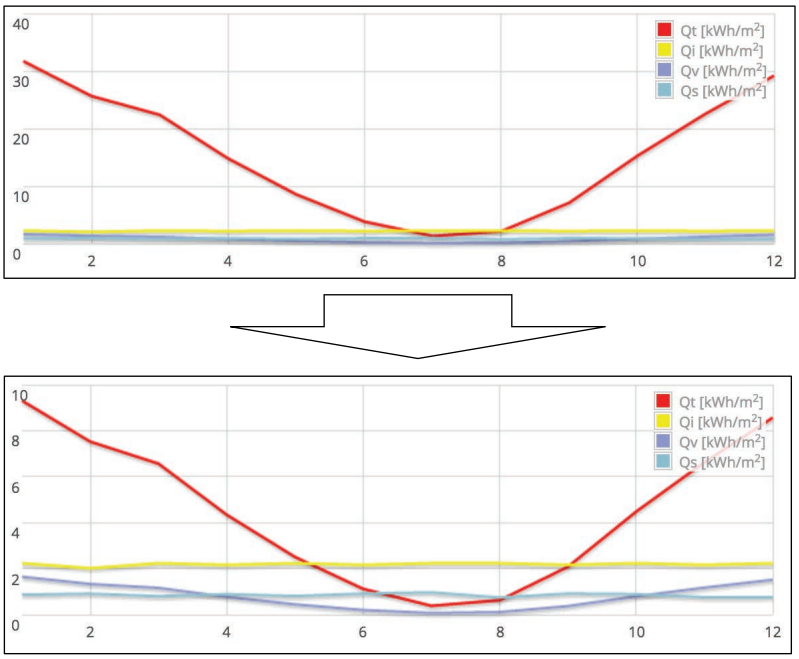

Figure 6.6: Detailed monthly energy demands before and after applied U-value changes.

6.3 Retrofit Module

Based on the results of the energy demand assessment and the calculated necessary optimizations, the retrofit module exposes the methods described in Chapter 5. The area that has to be considered for retrofit and the U-Value difference that should be achieved, are the simulation parameters that are used by the simulation service to calculate the retrofit scenario.

In the given case the interface is populated with the available product groups (Insulation Material, Plaster Material) and their respective products (Mineral Wool, EPS, Aerogel, Hemp Fiber, Silicate Plaster) that are included in the product ontology. As the assessment cycle is started, the ontology is used to calculate the necessary thickness of the material (Equation 5.2), and the material and labor costs resulting in a cost report of the completed retrofit cycle (Figure 6.8).

The extensibility of the assessment library is shown by the following extension. Based on the retrofit scenario result, the amortisation duration is calculated by an amortisation plugin. The manually specified energy

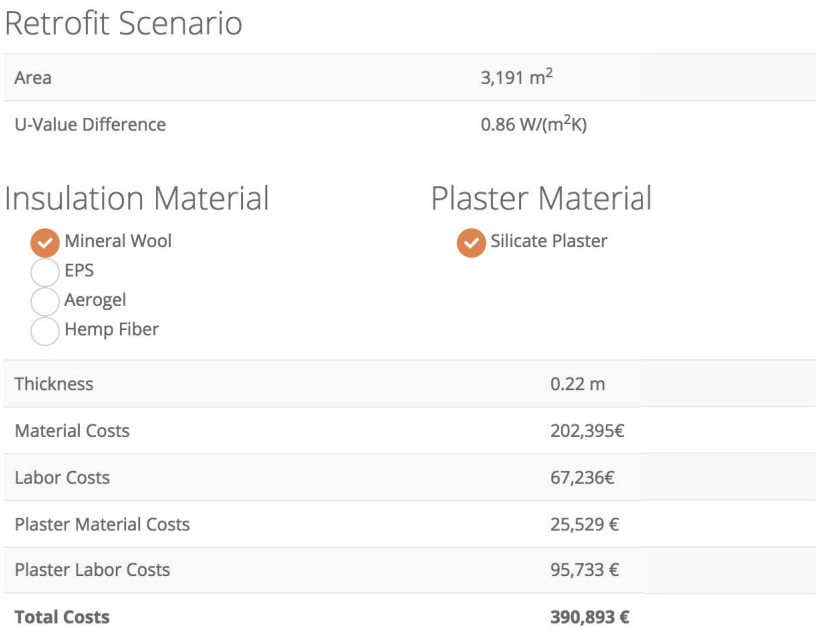

Figure 6.7: Calculated retrofit scenario.

Figure 6.8: Amortisation duration of the retrofit scenario.

price (0.13 EUR per kWh) and a customizable interest rate (5 %) are used to calculate the amortisation of the retrofit scenario. After 13 years, the investment costs in this example are amortised.

6.4 Services

Within the framework a couple of services are available that can be used to access certain logic on the client side. To implement the proposed web-application, ANGULARJS (Google 2015), a javascript framework was used. Figure 6.9 shows the angular services that contain the client-

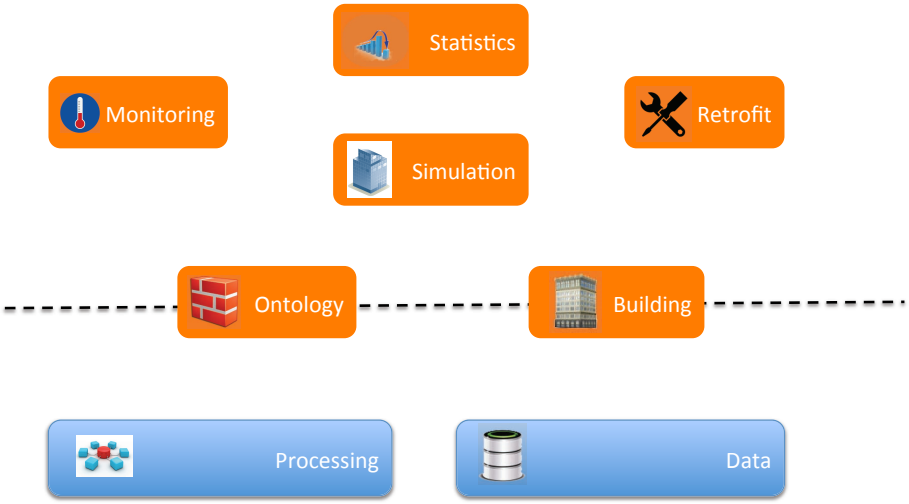

Figure 6.9: Overview of available services.

side logic, discussed in previous chapters. The building service loads buildings and monitoring data from the server and holds the active building instance. The ontology service handles all access operations on the building product ontology.

The monitoring service can contain logic to manipulate monitoring data and is responsible to analyse and visualize the zones and datapoints. The statistics service calculates building related statistics, such as exposed wall element area.

The simulation service entails all calculation methods that are currently implemented (e.g. monthly/ yearly GIS-based energy demand assessment). The retrofit service is used to calculate the retrofit costs and amortisation duration.

On the server side, two services allow access to the urban monitoring toolkit, as it has been discussed in Chapter 3. The processing service exposes data processing methods, such as periodization of measurements, but also data access to third party monitoring systems (e.g. MOST). Moreover, the data service allows access to raw data that is stored inside the urban monitoring toolkit.

6.5 IT infrastructure

For development purposes, virtual machines are used to deploy the toolkit. VirtualBox 2012 is used as the virtualization technology. Different virtual machines are used for the different toolkit modules. These are:

- **Database**

  The persistence-module is installed on a Debian/Linux system. Currently, the MySQL (version 5.7.9) metadata-module and the data-module with a Cassandra cluster (version 3.0.0) are installed on the same virtual machine. New Cassandra nodes can effortlessly be installed on other servers and added to the cluster.

- **Data abstraction**

  The data abstraction layers (service layer and service adapter) and the BMS business logic, interfaces to third-party monitoring systems (e.g. MOST) are deployed on an Apache Tomcat/ TomEE Java Application Server that supports the Java Enterprise Edition.

- **Virtual Datapoints**

  Currently, virtual datapoints are running on a Debian/Linux system.

- **Cassandra Nodes**

  At the moment, only one Cassandra Master Node is active. New nodes can easily be added on new virtual machines, as Cassandra handles the data replication automatically via an internal protocol.

The server that is used for VirtualBox provides 12 GByte of memory and eight Intel Xeon CPU cores. Stress test, which can be applied during development or building data import, can be realized with an additional virtual machine server with 24 GByte of RAM. The productive system is regularly backed up on an independent computer. Figure 6.10 shows the web-interface that is used to interact with the server system running all productive virtual machines.

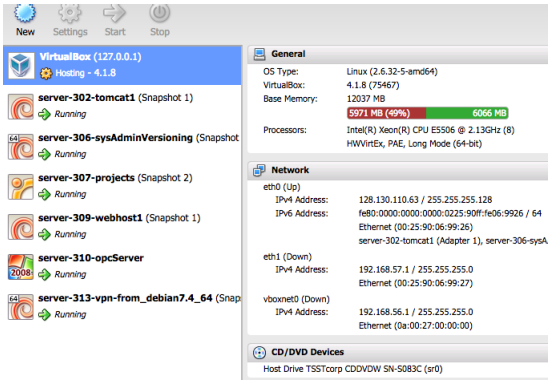

Figure 6.10: Web-Interface of the productive virtual machine server.

6.6 Summary

This chapter described the prototypical implementation of the proposed framework. In detail, the modules of a web-application are described that expose the logic of the proposed urban monitoring toolkit's core. A monitoring module allows to visualize zones and datapoints from various buildings, but also from stand-alone sensors (e.g. weather station). An energy demand assessment module is used to calculate GIS-based energy certificates for a selected building and to display statistics about the entity. The retrofit module utilizes a building product ontology to calculate retrofit scenarios.

CHAPTER 7

Conclusion

The presented concept of urban monitoring and simulation builds upon a holistic view of urban features. A comprehensive assessment of energy demand on an urban scale with various methods can support the decision finding processes of various stakeholders. For instance, the efficient management and organization of grids requires not only infrastructure but also a reliable analysis of energy demand and supply with high temporal and spatial resolution. While many domains (power stations, industry, mobility) offer sufficient data, the same cannot be said about the built environment. Comprehensive demand and supply data can offer valuable input for smart grid applications. This data can be generated and maintained by monitoring systems (Building Monitoring Systems BMS, Energy Management and Control Systems EMCS, Smart Meters) that often lack interoperability.

The urban environment produces massive amounts of data, not all of them related to buildings, but nevertheless influential (Pang et al. 2012). For instance, the integration of weather forecasts and historical weather data into a building's EMCS as well as the utilization as base data for light and thermal simulation has been documented in Schuss et al. (2013) and Saipi et al. (2013).

The work presented in this thesis provides a first step towards a generic urban data handling and simulation concept. Based on the requirements of the presented monitoring and simulation cases, a detailed

implementation concept was developed within this thesis. The procedure documented in this work can be used to support methods that deal with urban simulation and validation of energy flows in entire city districts.

7.1 Urban Monitoring

The proposed toolkit integrates various approaches to create a multi-purpose web-based urban monitoring system. By using the proposed toolkit, the discussed urban monitoring and simulation scenarios can be realized. Upon the urban monitoring toolkit, an application was developed that allows the analysis of building performance and retrofit for entire neighborhoods. The proposed toolkit enables users to couple individual building monitoring systems and sensors within a single application to serve certain urban applications (e.g. large-scale energy demand assessment).

The present work describes the efforts to scale a generic building monitoring system and the underlying data structure to support a distributed urban data exchange process. This process focuses on two main parts:

- building a highly generic data collection that offers an extensible and flexible data model
- developing a distributed application that offers standardized communication between physically distinct deployed modules.

The ideas and concepts presented in this work should be understood as platform-independent and provide a first step towards the realization of an urban data warehouse. The developed hybrid data store that includes both relational and non-relational components will be tested against the well established relational base system in terms of both reliability and performance. This will be realized not only by running extensive database tests, but also by simultaneously deploying both systems in a project environment.

Beside the implemented approaches to optimize data collection, future work should focus on data distribution. Connecting third party data sources is still rudimentarily available. One possible research topic could focus on developing support for other domains that require big data (e.g. traffic, public administration, health service, weather service, telecommunication and social networks), as suggested by Xu et al. (2015).

Another critical point is the integration of new sensors into the environment. Currently, sensor integration requires the implementation of *Connectors* that have been introduced by Zach and Mahdavi (2012). For instance, Yao et al. (2015b) introduced an IoT system to seamlessly integrate data from real world processes into virtual equivalents and vice versa. Critical research regards the seamless information access and exchange between physical and virtual realities. At the moment, our building monitoring data often lacks a specific social context. Typically, a context is established by a location (e.g. zone). Future research could address the integration of humans' social context into the analysis of building monitoring data. This could lead to valuable insights into the optimization potential of buildings, as people operate and live in buildings and thus are an important factor in building performance.

7.2 GIS-based Energy Demand Assessment

Besides the development efforts regarding the urban monitoring toolkit, this thesis presented an approach to (i) implement a simplified mathematical model to calculate energy certificates based on a rudimentary building model and (ii) to derive these building models from geographic input data (gBIM). Applying the procedure to a set of approximately 900 buildings in the city centre of Vienna showed promising results.

Specifically, buildings with a homogenous, non-complex building footprint and geometry are imported and processed correctly. The best results showed an error of $< 1\%$ when compared to the respective energy certificates. Nevertheless, there where deviations of up to 40% found that originated from complex geometries and roof structures. Currently, the

proposed framework does not consider roof geometries at all (flat roofs are assumed). Also, retrofit solutions are not included in the process, as GIS data does not offer such information.

Based on the promising results of the energy demand assessment, a more extensive analysis with an increased and more feasible sample size should be conducted.

7.3 Urban Retrofit Scenarios

Beside the GIS-based calculation of energy demands, the proposed building models are used to calculate retrofit scenarios. A rudimentary building product ontology is provided that demonstrates the feasibility of semantic web technologies for building-related applications. A simple model that depends on exposed wall element areas and number of building storeys is used to calculate the retrofit costs.

7.4 Future Research

Next steps focus on the (i) optimization of the gBIMs, (ii) data verification, (iii) thorough documentation and specification of gBIM, (iv) enhancement of the control sample and (v) the application and testing of gBIMs in a research project. The sample of 20 buildings is not sufficient to run a representative analysis, but offers a first starting point to test the quality of the proposed method.

The building product ontology must be extended in the future, regarding more classes, properties and instances. If a more comprehensive ontology is available, a mechanism to automatically import new building products could be developed. For instance, building materials can be classified by global warming potential and not just material costs.

The access performance of the database must be improved. Currently, all buildings are loaded regarding their presumed construction period. This process must be restricted to the visible map extend.

Currently, the energy demand assessment calculation method only focuses on monthly and yearly settings. It is planned to integrate a more

dynamic periodization of the calculation methods (e.g. energy demand per day). This also requires more accurate weather data, and not just the default climate settings.

As it can be seen in Figure 1.1, this thesis only covered two of the four possible urban monitoring and simulation scenarios. Future work can focus on the optimization of gBIMs to support third party simulation tools. Currently, an effort to export gBIMs as one-zone EnergyPlus models is being developed. Also the calibration of of the energy demands with monitoring measurements is a point that has to be addressed in the future.

As it has been discussed in the previous chapters, one building monitoring system and one sensor are connected to the proposed toolkit. This number can be increased. It is planned to include two more buildings that also operate MOST.

Beside the discussed future development fields, it is intended to test the toolkit within the context of a research project.

Publications

The following section gives an overview of the publications and reports that contain various portions of the present contribution and are partly included in this work.

Ghiassi, N., Glawischnig, S., Pont, U., and Mahdavi, A. (2014). “Toward a Data-Driven Performance-Guided Urban Decision-Support Environment”. In: *Information and Communication Technology Second IFIP TC5/8 International Conference ICT-EurAsia 2014 Bali*, p. 12.

Glawischnig, S. (2013). “Open Source Web-based Interaction with 3-dimensional Building Models via Human Interaction Devices - a Usability Study”. PhD thesis. Vienna University of Technology.

Glawischnig, S., Appel, R., Zach, R., and Mahdavi, A. (2012). “Recent advances in the development of MOST: an open source, wendor and technology independet toolkit for building monitoring, data prepro-

cessing and visualization.” In: *ICEBO - The International Conference for Enhanced Building Operations.* P. 8.

Glawischnig, S., Appel, R., Zach, R., and Mahdavi, A. (2013). “Usability Testing of Human Interface Devices For Building Information Systems”. In: *Interfaces and Human Computer Interaction 2013*, p. 8.

Glawischnig, S., Hammerberg, K., Vuckovic, M., Kiesel, K., and Mahdavi, A. (2014a). “A case study of geometry-based automated calculation of microclimatic attributes”. In: *Proceedings of the 10th European Conference on Product and Process Modelling*. Ed. by A. Mahdavi, B. Martens, and R.J. Scherer. Taylor and Francis - Balkema, 1/1/Boca Raton|London|New York|Leiden, pp. 231–236.

Glawischnig, S., Hofstätter, H., Bräuer, R., and Mahdavi, A. (2014b). “Bridging RDBMS and NoSQL to build a high-performance and scalable storage engine for building information systems”. In: *Proceedings of the 10th European Conference on Product and Process Modelling (ECPPM2014)*, pp. 231–236.

Glawischnig, S., Hofstätter, H., and Mahdavi, A. (2014c). “A Distributed Generic Data Structure for Urban Level Building Data Monitoring”. In: *Information and Communication Technology* 8407, p. 9.

— (2014d). “Application of Web-Technologies in Building Information Systems: the case of the Monitoring System Toolkit (MOST)”. In: *2nd ICAUD - Proceedings*, p. 7.

Glawischnig, S., Kiesel, K., and Mahdavi, A. (2014e). “Feasibility analysis of open-government data for the automated calculation of the microclimatic attributes of Urban Units of Observation in the city of Vienna”. In: *2nd ICAUD - Proceedings*, p. 6.

Glawischnig, S. and Mahdavi, A. (2013). “Human Interface Devices and Building Information Systems - A Usability Study”. In: *IADIS International Journal on WWW/Internet* 11.2, pp. 129–142.

— (2015). “Automated generation of topological building models from open government GIS-data for urban-level building analysis”. In: *Manuscript in preparation*.

Glawischnig, S., Schuss, M., and Mahdavi, A. (2015a). “Generating Energy Certificates from GIS-Data”. In: *Manuscript submitted for publication*.

Glawischnig, S., Tauber, C., and Mahdavi, A. (2015b). *Auto_Cal: Monitoringgestützte automatische Modellkalibrierung für Gebäudesteuerung und –diagnose*. Tech. rep. Department of Building Physics and Building Ecology, Technical University of Vienna.

Mahdavi, A., Glawischnig, S., and Ghiassi, N. (2016). “Urban Energy Computing: a Multi-Layered Approach”. In: *accepted for publication*.

Pont, U., Glawischnig, S., and Mahdavi, A. (2014). “AUTOCERT: A web-based approach for heating demand calculations for Building Performance Evaluation and Optimization”. In: *2nd ICAUD - Proceedings*, p. 8.

Zach, R., Glawischnig, S., Appel, R., Weber, J., and Mahdavi, A. (2012a). “Building data visualization using the open-source MOST framework and the GoogleWeb Toolkit”. In: *eWork and eBusiness in Architecture, Engineering and Construction*, pp. 747–752.

Zach, R., Glawischnig, S., Hönisch, M., Appel, R., and Mahdavi, A. (2012b). “MOST: An open-source, vendor and technology independent toolkit for building monitoring, data preprocessing, and visualization”. In: *eWork and eBusiness in Architecture, Engineering and Construction*, pp. 97–103.

Zach, R., Glawischnig, S., and Mahdavi, A. (2013a). “Advanced building data visualization using the Monitoring System Toolkit”. In: *CLIMA 2013 - 11th REHVA World Congress and the 8th International Conference on Indoor Air Quality, Ventilation and Energy Conservation in Buildings*, p. 9.

Zach, R., Hofstätter, H., Glawischnig, S., and Mahdavi, A. (2013b). “Incorporation of Run-Time Simulation-Powered Virtual Sensors in Building Monitoring Systems”. In: *BS2013 - Building Simulation 2013*. Chambery, France, pp. 2083–2089. DOI: ISBN:978-2-7466-6294-0.

— (2013c). “Incorporation of run-time simulation-powered virtual sensors in building monitoring systems”. In: *Building Simulation 2013-*

13th International Conference of the International Building Performance Simulation Association, pp. 2083–2089.

This section also contains papers that are still in print and not yet published.

List of Figures

List of Tables

APPENDIX A

Ontology Prototype

```
<?xml version="1.0"?>


<!DOCTYPE Ontology [
    <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
    <!ENTITY xml "http://www.w3.org/XML/1998/namespace" >
    <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
    <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
]>


<Ontology xmlns="http://www.w3.org/2002/07/owl#"
     xml:base="http://www.semanticweb.org/sg/ontologies/2015/10/untitled-o
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:xml="http://www.w3.org/XML/1998/namespace"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
     ontologyIRI="http://www.semanticweb.org/sg/ontologies/2015/10/untitled
    <Prefix name="" IRI="http://www.w3.org/2002/07/owl#"/>
    <Prefix name="owl" IRI="http://www.w3.org/2002/07/owl#"/>
    <Prefix name="rdf" IRI="http://www.w3.org/1999/02/22-rdf-syntax-ns#"/>
```

```
<Prefix name="xsd" IRI="http://www.w3.org/2001/XMLSchema#"/>
<Prefix name="rdfs" IRI="http://www.w3.org/2000/01/rdf-schema#"/>
<Declaration>
    <Class IRI="#Brickwork_Mortar"/>
</Declaration>
<Declaration>
    <Class IRI="#Building"/>
</Declaration>
<Declaration>
    <Class IRI="#EPS_Insulation_Panel"/>
</Declaration>
<Declaration>
    <Class IRI="#Exterior_Rendering"/>
</Declaration>
<Declaration>
    <Class IRI="#Hemp_Insulation"/>
</Declaration>
<Declaration>
    <Class IRI="#Insulation"/>
</Declaration>
<Declaration>
    <Class IRI="#Mineral"/>
</Declaration>
<Declaration>
    <Class IRI="#Mineral_Wool_Insulation"/>
</Declaration>
<Declaration>
    <Class IRI="#Product"/>
</Declaration>
<Declaration>
    <Class IRI="#Renewable"/>
</Declaration>
<Declaration>
```

```
    <Class IRI="#Silicate_Plaster_Exterior"/>
</Declaration>
<Declaration>
    <Class IRI="#Synthetic"/>
</Declaration>
<Declaration>
    <ObjectProperty IRI="#compatibleWith"/>
</Declaration>
<Declaration>
    <ObjectProperty IRI="#compatibleWithForUValue"/>
</Declaration>
<Declaration>
    <ObjectProperty IRI="#labor"/>
</Declaration>
<Declaration>
    <ObjectProperty IRI="#material"/>
</Declaration>
<Declaration>
    <ObjectProperty IRI="#pricePerUnit"/>
</Declaration>
<Declaration>
    <ObjectProperty IRI="#productSpecific"/>
</Declaration>
<Declaration>
    <ObjectProperty IRI="#supportsInsulationMaterial"/>
</Declaration>
<Declaration>
    <ObjectProperty IRI="#supportsMortar"/>
</Declaration>
<Declaration>
    <ObjectProperty IRI="#thermalConductivity"/>
</Declaration>
<Declaration>
```

```
    <ObjectProperty IRI="#unit"/>
</Declaration>
<Declaration>
    <DataProperty IRI="#labor"/>
</Declaration>
<Declaration>
    <DataProperty IRI="#material"/>
</Declaration>
<Declaration>
    <DataProperty IRI="#pricePerUnit"/>
</Declaration>
<Declaration>
    <DataProperty IRI="#thermalConductivity"/>
</Declaration>
<Declaration>
    <DataProperty IRI="#unit"/>
</Declaration>
<Declaration>
    <NamedIndividual IRI="#Default_EPS"/>
</Declaration>
<SubClassOf>
    <Class IRI="#Brickwork_Mortar"/>
    <Class IRI="#Product"/>
</SubClassOf>
<SubClassOf>
    <Class IRI="#EPS_Insulation_Panel"/>
    <Class IRI="#Synthetic"/>
</SubClassOf>
<SubClassOf>
    <Class IRI="#Exterior_Rendering"/>
    <Class IRI="#Brickwork_Mortar"/>
</SubClassOf>
<SubClassOf>
```

```
        <Class IRI="#Hemp_Insulation"/>
        <Class IRI="#Renewable"/>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#Insulation"/>
        <Class IRI="#Product"/>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#Mineral"/>
        <Class IRI="#Insulation"/>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#Mineral_Wool_Insulation"/>
        <Class IRI="#Mineral"/>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#Product"/>
        <Class IRI="#Building"/>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#Renewable"/>
        <Class IRI="#Insulation"/>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#Silicate_Plaster_Exterior"/>
        <Class IRI="#Exterior_Rendering"/>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#Synthetic"/>
        <Class IRI="#Insulation"/>
    </SubClassOf>
    <ClassAssertion>
        <Class IRI="#EPS_Insulation_Panel"/>
```

```
        <NamedIndividual IRI="#Default_EPS"/>
    </ClassAssertion>
    <ObjectPropertyAssertion>
        <ObjectProperty IRI="#labor"/>
        <NamedIndividual IRI="#Default_EPS"/>
        <NamedIndividual IRI="#Default_EPS"/>
    </ObjectPropertyAssertion>
    <ObjectPropertyAssertion>
        <ObjectProperty IRI="#material"/>
        <NamedIndividual IRI="#Default_EPS"/>
        <NamedIndividual IRI="#Default_EPS"/>
    </ObjectPropertyAssertion>
    <ObjectPropertyAssertion>
        <ObjectProperty IRI="#pricePerUnit"/>
        <NamedIndividual IRI="#Default_EPS"/>
        <NamedIndividual IRI="#Default_EPS"/>
    </ObjectPropertyAssertion>
    <ObjectPropertyAssertion>
        <ObjectProperty IRI="#thermalConductivity"/>
        <NamedIndividual IRI="#Default_EPS"/>
        <NamedIndividual IRI="#Default_EPS"/>
    </ObjectPropertyAssertion>
    <ObjectPropertyAssertion>
        <ObjectProperty IRI="#unit"/>
        <NamedIndividual IRI="#Default_EPS"/>
        <NamedIndividual IRI="#Default_EPS"/>
    </ObjectPropertyAssertion>
    <DataPropertyAssertion>
        <DataProperty IRI="#labor"/>
        <NamedIndividual IRI="#Default_EPS"/>
        <Literal datatypeIRI="&xsd;float">2.0</Literal>
    </DataPropertyAssertion>
    <DataPropertyAssertion>
```

```
        <DataProperty IRI="#material"/>
        <NamedIndividual IRI="#Default_EPS"/>
        <Literal datatypeIRI="&xsd;float">1.13</Literal>
    </DataPropertyAssertion>
    <DataPropertyAssertion>
        <DataProperty IRI="#thermalConductivity"/>
        <NamedIndividual IRI="#Default_EPS"/>
        <Literal datatypeIRI="&xsd;float">0.035</Literal>
    </DataPropertyAssertion>
    <DataPropertyAssertion>
        <DataProperty IRI="#unit"/>
        <NamedIndividual IRI="#Default_EPS"/>
        <Literal datatypeIRI="&xsd;string">1 m^−3</Literal>
    </DataPropertyAssertion>
    <SubObjectPropertyOf>
        <ObjectProperty IRI="#compatibleWith"/>
        <ObjectProperty IRI="#productSpecific"/>
    </SubObjectPropertyOf>
    <SubObjectPropertyOf>
        <ObjectProperty IRI="#compatibleWithForUValue"/>
        <ObjectProperty IRI="#productSpecific"/>
    </SubObjectPropertyOf>
    <SubObjectPropertyOf>
        <ObjectProperty IRI="#labor"/>
        <ObjectProperty IRI="#pricePerUnit"/>
    </SubObjectPropertyOf>
    <SubObjectPropertyOf>
        <ObjectProperty IRI="#material"/>
        <ObjectProperty IRI="#pricePerUnit"/>
    </SubObjectPropertyOf>
    <SubObjectPropertyOf>
        <ObjectProperty IRI="#supportsInsulationMaterial"/>
        <ObjectProperty IRI="#productSpecific"/>
```

```
    </SubObjectPropertyOf>
    <SubObjectPropertyOf>
        <ObjectProperty IRI="#supportsMortar"/>
        <ObjectProperty IRI="#productSpecific"/>
    </SubObjectPropertyOf>
    <InverseObjectProperties>
        <ObjectProperty IRI="#supportsInsulationMaterial"/>
        <ObjectProperty IRI="#supportsMortar"/>
    </InverseObjectProperties>
    <SubDataPropertyOf>
        <DataProperty IRI="#labor"/>
        <DataProperty IRI="#pricePerUnit"/>
    </SubDataPropertyOf>
    <SubDataPropertyOf>
        <DataProperty IRI="#material"/>
        <DataProperty IRI="#pricePerUnit"/>
    </SubDataPropertyOf>
</Ontology>



<!-- Generated by the OWL API (version 3.5.1) http://owlapi.sourceforge.ne
```

Bibliography

Apache (2015a). *Apache Cassandra, http://cassandra.apache.org/*.

— (2015b). *Apache Hadoop, https://hadoop.apache.org/*.

ASI (2011). *ÖNORM B8110-5: Thermal insulation in building construction – Part 5: Model of climate and user profiles.* Tech. rep. Austrian Standards Institute.

— (2014). *Thermal insulation in building construction – Part 6: Principles and verification methods – Heating demand and cooling demand – national application, national specifications and national supplements to ÖNORM EN ISO 13790*. Tech. rep. Austrian Standards Institute.

Autodesk (2015). *Project Dasher, http://www.autodeskresearch.com/projects/dasher*.

Baldauf, M., Dustdar, S., and Rosenberg, F. (2007). "A survey on context-aware systems". In: *International Journal of Ad Hoc and Ubiquitous Computing* 2.4, pp. 263–277.

Baubuch (2013). *Baubuch 2013, http://www.baubook.at*.

Beckett, Dave (2004). *RDF/XML Syntax Specification (Revised)*. W3C Recommendation. http://www.w3.org/TR/2004/REC-rdf-syntax-grammar-20040210/. W3C.

BEV (2006). *Spatial Reference EPSG:31256*. Tech. rep. A. Bundesamt für Eich- und Vermessungswesen. URL: http://www.bev.gv.at/.

BIMserver (2014). *Open Source Building Information Modelserer, http://bimserver.org/*. Accessed: 01.07.2014.

BIMsurfer (2014). *Open Source WebGL viewer for IFC models, http://bimsurfer.org/*. Accessed: 01.07.2014.

Brown, M.J., Grimmond, C., Betham, S., and Ratti, C. (2001). "Comparison of Methodologies for Computing Sky View Factor in Urban Environments". In: *Proceedings of the 2001 International Symposium on*

Environmental Hydraulics. International Association for Hydro-Environment Engineering and Research.

Commons, Creative (2014). *Creative Commons Licence (CC BY 3.0 AT)*. URL: https://creativecommons.org/licenses/by/3.0/at/deed.de.

Dam, F., Mandl, W., and König, H. (2014). *SIRADOS Kalkulationsatlas 2014 für Roh- und Ausbau im Altbau*. WEKA.

Davis (2015). *Davis Vantage Pro2, http://www.davisnet.com/weather/products/vantage-pro-professional-weather-stations.asp*.

Dustdar, S., Vögler, M., Sehic, S., Qanbari, S., Nastic, S., and Truong, H. (2014). "The Internet of Things Meets Cloud Computing in Smart Cities". In: *Bridges* 41.

EU (2007). "Directive 2007/2/EC of the European Parliament and of the Council". In: *Official Journal of the European Union*.

— (2010). "Directive 2010/31/EU of the European Parliament and of the Council". In: *Official Journal of the European Union*.

Fenz, S., Heurix, J., Neubauer, T., Tjoa, A., Ghiassi, N., Pont, U., and Mahdavi, A. (2014). "SEMERGY.net - automatically identifying and optimizing energy-efficient building designs". In: *Computer Science - Research and Development*, pp. 1–6.

FFG (2015). *The Austrian Research Promotion Agency, https://www.ffg.at/en*.

GenOpt (2014). *Generic Optimization Program, http://http://simulationresearch.lbl.gov/GO/*. Accessed: 01.07.2014.

Giridharan, R., Ganesan, S., and Lau, S.S.Y. (2004). "Daytime urban heat island effect in high-rise and high-density residential developments in Hong Kong". In: *Energy and Buildings* 36, pp. 525–534.

Glawischnig, S. (2013). "Open Source Web-based Interaction with 3-dimensional Building Models via Human Interaction Devices - a Usability Study". PhD thesis. Vienna University of Technology.

Glawischnig, S., Appel, R., Zach, R., and Mahdavi, A. (2012). "Recent advances in the development of MOST: an open source, wendor and technology independet toolkit for building monitoring, data prepro-

cessing and visualization." In: *ICEBO - The International Conference for Enhanced Building Operations.* P. 8.

Glawischnig, S., Hammerberg, K., Vuckovic, M., Kiesel, K., and Mahdavi, A. (2014a). "A case study of geometry-based automated calculation of microclimatic attributes". In: *Proceedings of the 10th European Conference on Product and Process Modelling*. Ed. by A. Mahdavi, B. Martens, and R.J. Scherer. Taylor and Francis - Balkema, 1/1/Boca Raton|London|New York|Leiden, pp. 231–236.

Glawischnig, S., Hofstätter, H., and Mahdavi, A. (2014b). "A Distributed Generic Data Structure for Urban Level Building Data Monitoring". In: *Information and Communication Technology* 8407, p. 9.

Glawischnig, S., Kiesel, K., and Mahdavi, A. (2014c). "Feasibility analysis of open-government data for the automated calculation of the microclimatic attributes of Urban Units of Observation in the city of Vienna". In: *2nd ICAUD - Proceedings*, p. 6.

Glawischnig, S. and Mahdavi, A. (2013). "Human Interface Devices and Building Information Systems - A Usability Study". In: *IADIS International Journal on WWW/Internet* 11.2, pp. 129–142.

Google (2015). *AngularJS - Superheroic JavaScript MVW Framework,https://angularjs.org/*.

Graubner, C. and Hüske, K. (2003). *Nachhaltigkeit im Bauwesen*. Berlin, Germany: Ernst und Sohn.

Grimmond, C.S.B. and Oke, T.R. (1999). "Heat Storage in Urban Areas: Local-Scale Observations and Evaluation of a Simple Model". In: *Journal of Applied Meteorology* 38.7, pp. 922–940.

Gruber, T. (1993). "A Translation Approach to Portable Ontology Specifications". In: *Knowledge Acquisition* 5, pp. 199–220.

Harris, Steven and Seaborne, Andy (2013). *SPARQL 1.1 Query Language*. W3C Recommendation. http://www.w3.org/TR/2013/REC-sparql11-query-20130321/. W3C.

Hecht, R. and Jablonski, S. (2011). "NoSQL evaluation: A use case oriented survey". In: *Proc. CSC 2011*. Hong Kong, Rep. China, pp. 336–341. ISBN: 9781457716355.

Holmer, B., Postgard, U., and Eriksson, M. (2001). “Sky view factors in forest canopies calculated with IDRISI”. In: *Theoretical and Applied Climatology* 68, pp. 33–40.

IPCC (2014). *Climate Change 2014: Synthesis Report*. Tech. rep. Intergovernmental Panel on Climate Change.

ISO (2012). *Energy performance of buildings – Calculation of energy use for space heating and cooling*. ISO. International Organization for Standardization.

Johnson, G.T. and Watson, I.D. (1984). “The determination of view-factors in urban canyons”. In: *Journal of Applied Meteorology* 23, pp. 329–335.

Jung, M. (2015). “An integration middleware for the Internet of Things”. PhD thesis. TU Wien.

Jung, M., Chelkal, J., Schober, J., Kastner, W., Zhou, L., and Nam, G. (2014). “IoTSyS: an integration middleware for the Internet of Things.” In: *Proceedings of the 4th International Conference on the Internet of Things*.

Khan, A. and Hornbæk, K. (2011). “Big Data from the Built Environment”. In: *Proceedings of the 2Nd International Workshop on Research in the Large*. LARGE ’11. New York, NY, USA: ACM, pp. 29–32. ISBN: 9781450309240. URL: http://doi.acm.org/10.1145/2025528.2025537.

Kiesel, K. (2015). “The influence of the urban microclimate on the thermal performance of buildings”. PhD thesis. Karlsplatz 13, 1040 Vienna: Vienna University of Technology.

Kiesel, K., Vuckovic, M., and Mahdavi, A. (2013). “Representation of weather conditions in building performance simulation: A case study of microclimatic variance in central europe”. In: *13th International Conference of the International Building Performance Simulation Association*.

Klein, D., Tran-Gia, P., and Hartmann, M. (2013). “Big Data”. In: *Informatik-Spektrum* 36.3, pp. 319–323. URL: http://dx.doi.org/10.1007/s00287-013-0702-3.

KNX (2016). *KNX Association*. Accessed: 20.01.2016. URL: http://www.knx.org.

Laney, D. (2001). *3D Data Management: Controlling Data Volume, Velocity and Variety*. Tech. rep. META Group. URL: http://blogs.gartner.com/doug-laney/files/2012/01/ad949-3D-Data-Management-Controlling-Data-Volume-Velocity-and-Variety.pdf.

Leavitt, N. (2010). "Will NoSQL Databases will Live Up to Their Promise?" In: *IEEE Computer* 43.2, pp. 12–14.

Lee, S.H. and Park, S.U. (2008). "A vegetated urban canopy model for meteorological and environmental modeling". In: *Boundary-Layer Meteorology* 126.1, pp. 73–102.

Lindberg, F. (2007). "Modeling the urban climate using a local governmental geo-database". In: *Meteorological Applications* 14, pp. 263–273.

Mahdavi, A. (2012). "From Building Physics to Urban Physics". In: *Stadt: Gestalten*. Springer Vienna. Chap. 34, pp. 181–186.

Mahdavi, A., Kiesel, K., and Vuckovic, M. (2013). "A framework for the evaluation of urban heat island mitigation measures". In: *SB13*.

Mahdavi, A., Pont, U., Ghiassi, N., Shayeganfar, F., Fenz, S., Heurix, J., Anjomshoaa, A., and Tjoa, A. (2015). "Performance-based building design and retrofit optimization: The SEMERGY approach". In: *Digital Opportunities in Architecture* 35, pp. 49–57.

Mayer-Schonberger, V. and Cukier, K. (2013). *Big Data: A Revolution That Will Transform How We Live, Work, and Think*. Eamon Dolan / Houghton Mifflin Harcourt. ISBN: 0544002695.

M-Bus (2016). *M-Bus*. Accessed: 20.01.2016. URL: www.m-bus.com.

Memon, R.A., Leung, D.Y.C., and Chunho, L. (2007). "A review on the generation, determination and mitigation of Urban Heat Island". In: *Journal of Environmental Sciences* 20, pp. 120–128.

MOST (2012). *Monitoring System Toolkit, http://most.bpi.tuwien.ac.at*. Accessed: 01.07.2014.

Neo4j (2015). *Neo4j - The worlds leading graph database, http://neo4j.com/*.

Nunez, M. and Oke, T.R. (1977). “The Energy Balance of an Urban Canyon”. In: *Journal of Applied Meteorology* 16, pp. 11–19.

oBIX (2016). *Open Building Information Xchange*. Accessed: 20.01.2016. URL: www.obix.org.

OiB (2015). *Energieeinsparung und Wärmeschutz*. Tech. rep. Österreichisches Institut für Bautechnik.

Oke, T.R. (1981). “Canyon geometry and the nocturnal urban heat island: comparison of scale model and field observations”. In: *J Climatol* 1, pp. 237–254.

— (1982). “The energy basis of the urban heat island”. In: *Quart. J.R. Met. Soc* 108, pp. 1–24.

Okman, L., Gal-Oz, N., Gonen, Y., Gudes, E., and Abramov, J. (2011). “Security Issues in NoSQL Databases”. In: *Proc. IEEE TrustCom-11/ IEEE ICESS-11/FCST-1*. Liverpool, UK, pp. 541–547. DOI: 978-1-4577-2135-9.

Pang, X., Wetter, M., Bhattacharya, P., and Haves, P. (2012). “A framework for simulation-based real-time whole building performance assessment”. In: *Building and Environment* 54, p. 8.

Parliament, European and European Council, the (2015). *INSPIRE*. Accessed: 03.12.2015. URL: http://inspire.ec.europa.eu/.

Pont, U. (2014). “A comprehensive approach to web.enabled optimization-based decision support in building design and retrofit”. PhD thesis. Vienna University of Technology.

Pont, U., Ghiassi, N., Fenz, S., Heurix, J., and Mahdavi, A. (2015). “SEMERGY: Application of Semantic Web Technologies in Performance-Guided Building Design Optimization”. In: *Journal of Information Technology in Construction* 20.Special Issue, pp. 107–120.

Pont, U. and Mahdavi, A. (2015). *Innovative Baustoffe, Bausysteme und Informationstechnologien für die Gebäudesanierung*.

Qanbari, S., Behinaein, N., Rahimzadeh, R., and Dustdar, S. (2015). “Gatica: Linked Sensed Data Enrichment and Analytics Middleware for IoT Gateways”. In: *Future Internet of Things and Cloud (FiCloud)*, pp. 38–43.

Radiance (2014). *A validated Lightning Simulation Tool, http://www.radiance-online.org/*. Accessed: 01.07.2014.

Richens, P. (1997). “Image processing for urban scale environmental modeling”. In: *Proc. 5th International IBPSA Conferences: Building Simulation 97*.

Saipi, N., Schuss, M., Pont, U., and Mahdavi, A. (2013). “Comparison of simulated and actual energy use of a hospital building in Austria.” In: *enviBUILD 2013 - Buildings and Environment*. Vol. 1, p. 8.

Santamouris, M. (2007). “Heat Island Research in Europe: The state of the art”. In: *Advances in Building Energy Research* 1, pp. 123–150.

Schuss, M., Tahmasebi, F., Vazifeh, E., and Mahdavi, A. (2013). “The influence of online weather forecast uncertainties on the performance of predictive zone controllers.” In: *enviBUILD 2013 - Buildings and Environment*. Ed. by Josef Hraska et al., p. 7.

SIMULTAN (2016). *Simultaneous planning environment for buildings in resilient, highly energy efficient and resource-efficient districts*. Accessed: 05.01.2016. URL: http://www.hausderzukunft.at/results.html/id7835.

SkyFoundry (2015). *Folio Database, http://skyfoundry.com/skyspark*.

Stewart, I.D. and Oke, T.R. (2012). “Local climate zones for urban temperature studies”. In: *Bulletin of the American Meteorological Society* 93.12, pp. 1879–1900.

Taha, H. (1997). “Urban climates and heat islands: albedo, evapo-transpiration, and anthroprogenic heat”. In: *Energy and Buildings* 25, pp. 99–103.

Tauber, C., Tahmasebi, F., Zach, R., and Mahdavi, A. (2014). “Automated simulation model calibration based on runtime building monitoring”. In: *Proc. ECPPM 2014*, accepted for publication.

Tudorica, B.G. and Bucur, C. (2011). “A comperison between several NoSQL databases with comments and notes”. In: *Proc. ProEduNet 2011*. Iasi, Romania, pp. 1–5. DOI: 978-1-4577-1233-3.

UIUC and LBNL (2007). *EnergyPlus Engineering Reference*.

Unger, J. (2004). “Intra-urban relationship between surface geometry and urban heat island: review and new approach”. In: *Climate Research* 27, pp. 253–264.

Vinoski, S. (2006). “Advanced message queuing protocol.” In: *Internet Computing, IEEE* 10.6, pp. 87–89.

Vretanos, P.A. (2005). *Web Feature Service Implementation Specification*. Tech. rep. Open Geospatial Consortium.

Weber, J., Zach, R., Tahmasebi, F., and Mahdavi, A. (2012). “Inclusion of user-related Monitoring Data in the run-time calibration of building performance simulation models”. In: *Proc. of Bausim 2012*. Berlin, Germany.

Wolosiuk, D., Ghiassi, G., Pont, U., Shayeganfar, F., Mahdavi, A., Fenz, S., Heurix, J., Anjomshoaa, A., and Tjoa, A. Min (2013). “Semergy: Performance-guided building design and refurbishment within a semantically augmented optimization environment.” In: *enviBUILD 2013 - Buildings and Environment*. Ed. by Josef Hraska.

Xu, X., Sheng, Q., Zhang, L., Fan, Y., and Dustdar, S. (2015). “Form Big Data to Big Service”. In: *Computer* 48.7, pp. 80–83.

Yao, L., Sheng, Q., Benatallah, B., Dustdar, S., Shemshadi, A., Wang, X., and Ngu, A. (2015a). “Up in the Air: When Homes Meet the Web of Things”. In: *arXiv preprint*.

Yao, L., Sheng, Q., and Dustdar, S. (2015b). “Web-Based Management of the Internet of Things”. In: *Internet Computing, IEEE* 19.4, pp. 60–67.

Zach, R. (2012). “An open-source, vendor and technology independent toolkit for building monitoring, data preprocessing, and visualization.” PhD thesis. Vienna University of Technology.

Zach, R., Hofstätter, H., Glawischnig, S., and Mahdavi, A. (2013a). “Incorporation of Run-Time Simulation-Powered Virtual Sensors in Building Monitoring Systems”. In: *BS2013 - Building Simulation 2013*. Chambery, France, pp. 2083–2089. DOI: ISBN:978-2-7466-6294-0.

— (2013b). “Incorporation of run-time simulation-powered virtual sensors in building monitoring systems”. In: *Building Simulation 2013-*

13th International Conference of the International Building Performance Simulation Association, pp. 2083–2089.

Zach, R. and Mahdavi, A. (2012). “MOST - designing a vendor and technology independent toolkit for building monitoring, data preprocessing, and visualization.” In: *Proceedings - First International Conference on Architecture and Urban Design*. Vol. 1. Epoka University Press.

Zach, R., Schuss, M., Bräuer, R., and Mahdavi, A. (2012). “Improving building monitoring using a data preprocessing storage engine based on MySQL”. In: *Proc. ECPPM 2012*. Reykjavik, Iceland, pp. 151–157. DOI: ISBN:978-0-415-62128-1.

Ziegler, S., Kastner, W., and Jung, M. (2013). “IoT6 - Moving to an IPv6-Based Future IoT”. In: *The Future Internet*, pp. 161–172.

Glawischnig Stefan

Education

2005–2008 **BSc**, *Carinthia University of Applied Sciences*, Villach.
Bachelor Studies *Geoinformation*

2008–2013 **Dipl.-Ing.**, *Technical University of Vienna*, Vienna.
Master Studies *Geoinformation and Cartography*

2010 **Short term exchange**, *Aalto University of Technology*, Helsinki.
Focus: spatial Algorithms

2013 **Doctoral College**, *Technical University of Vienna*, Vienna.
Environmental Informatics

2013-2017 **University Assistant**, *Technical University of Vienna*, Vienna.
Department of Building Physics and Building Ecology. *Director:* Ardeshir Mahdavi

Master thesis

title *Open Source Web-based Interaction with 3-dimensional Building Models via Human Interaction Devices- a Usability Study*

supervisors Andrew U. Frank, Ardeshir Mahdavi

Experience

Projects

2008 **Application Developer**, *Nationalpark Hohe Tauern*.
Geodata-Infrastructure, Biodiversitydatabase in cooperation with Haus der Natur Salzburg and Nationalpark Hohe Tauern.

2011–2012 **Web Development**, *FWF-Project*.
Development of a multi-tier application to monitor energy performance and occupancy data in buildings.

2012 **Mentor**, *Google Summer of Code*.
Web-based, 3D building visualization, building data export.

2013 **Developer**, *EU-Project*.
Urban Heat Island: Evaluation process of UHI phenomenon with open geometry data.

2013 **Mentor**, *Google Summer of Code*.
NoSQL solution for building data warehouses.

Meidlinger Hauptstraße 42-44 Top 71/72 – 1120 Wien – Austria
+43 (699) 1734 6019 • *stefan.glawischnig@gmx.at*

Project Management

2014–2015 **Auto_Cal**, *FFG-Project*.
Real-time calibration of thermal building models with monitoring data.

2015–2016 **E_PROFIL**, *FFG-Project*.
Large-scale energy demand assessment of entire building blocks.

Part-time work

2006– **Web-development Freelancer**.
Small - medium web projects.

2010 **Quality Assurance**, *dowee it solutions GmbH*.
Software development for international marine insurance consortium.

2010– **Open Source Development**.
Java, JavaScript.

2011–2013 **Project Assistant**.
Dept. of building physics and building ecology.

Self-employed

2012– **Android Mobile Applications**.
Fun apps, organizer

Languages

German Mother Tongue
English Very good

Computer skills

I know my way around a computer.

Interests

Sports Soccer, Squash, Swimming
Winter Sports Skiing/ Snowboarding Instructor
Qi Gong Instructor
Nordic Walking Instructor
Evening Class Education as an EU-Expert. Passed with excellence.
Communication Organization of Talk-evenings with the Akademie Wien.

Meidlinger Hauptstraße 42-44 Top 71/72 – 1120 Wien – Austria
+43 (699) 1734 6019 • stefan.glawischnig@gmx.at