
Object-Oriented Data Modeling and Warehousing to Support Urban Design

(The City of Makkah as a Case Study)

Nabeel A. Koshak

Ph.D. Dissertation

Submitted to the School of Architecture
of Carnegie Mellon University in partial fulfillment of
the requirements for the degree of Doctor of Philosophy

Computational Design
School of Architecture
Carnegie Mellon University

May 2002

Advisory Committee

Ulrich Flemming (Chair)

Professor
School of Architecture

Ömer Akin

Professor
School of Architecture

Christos Faloutsos

Professor
School of Computer Science

Wilpen Gorr

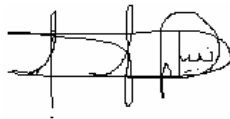
Professor
Heinz School of Public Policy & Management



I hereby declare that I am the author of this dissertation.

I authorize Carnegie Mellon University to lend this dissertation to other institutions or individuals for the purpose of scholarly research.

I further authorize Carnegie Mellon University to reproduce this dissertation by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

A handwritten signature in black ink, appearing to read 'Nabeel Koshak', written over a horizontal line.

Nabeel Koshak

Copyright © 2002 by Nabeel Koshak
All rights reserved

CARNEGIE MELLON UNIVERSITY

**School of Architecture
College of Fine Arts**

Dissertation

Submitted in Partial Fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

TITLE: *Object-Oriented Data Modeling and Warehousing to
Support Urban Design*

PRESENTED BY:

Nabeel A. Koshak


ACCEPTED BY ADVISORY COMMITTEE:



Ulrich Flemming Principal Advisor

5/8/02

DATE



Christos Faloutsos Advisor

5/8/02

DATE



Wilpen Gorr Advisor

5/8/02

DATE



Ömer Akin Advisor

5/14/02

DATE

CARNEGIE MELLON UNIVERSITY

**College of Fine Arts
School of Architecture**

Dissertation

Submitted in Partial Fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

TITLE: *Object-Oriented Data Modeling and Warehousing to
Support Urban Design*

PRESENTED BY *Nabeel A. Koshak*

ACCEPTED BY:

Martin Prekop
Martin Prekop Dean

06/14/02
DATE

Vivian Loftness
Vivian Loftness Head of School

MAY 23, 2003
DATE

Ulrich Flemming
Ulrich Flemming Principal Advisor

05/14/02
DATE

Abstract

All over the world, local authorities are moving towards managing and storing urban data in digital form. But the data storage devices used are heterogeneous and typically include relational database management systems (DBMS), GIS, and CAD files. As a result, data are present in different locations on different platforms and under different schemas. This poses a problem for software applications meant to support decision-making in urban design that require input from more than one data source.

This dissertation demonstrates how *data warehousing*—combined with *object-oriented data modeling*—is able to provide a general solution for this problem. Data warehousing is a technique initially developed for business applications, but is equally useful for urban design. The data warehouse constitutes a communication layer between the urban design applications and data sources. It makes the data available through a unified interface that hides the sources themselves and represents that data in terms of a general-purpose, preferably object-oriented, model. The data model is developed based on an analysis of the data needs of typical applications that support urban design and planning. Key components of the proposed warehouse architecture are *Data Access Objects* communicating with, *data extractors* able to handle queries expressed in terms of the data model, and *solvers* that detect contradictions in data and are able to perform data cleaning. This dissertation also describes a first prototype implementation of the data model and the data warehouse.

The test case used in this research to validate the object model and data warehouse architecture is the city of Makkah in Saudi Arabia, which faces significant urban design and planning challenges in connection with the annual pilgrimage that brings millions of visitors to the city. This research is sponsored by The Custodian of the Two Holy Mosques Institute for Hajj Research in Makkah, Saudi Arabia.

Acknowledgement

It is a great pleasure to acknowledge the assistance and help provided by those who care about this work. First, I would like to express my deepest love and gratitude for my parents, Mr. Abdulkadir Koshak and Mrs. Fawziah Abduljabbar. Their support, guidance, and advice have been the principal factors in getting me this far.

I have been very privileged to work with all four members of my advisory committee. It was a very enlightening and humbling experience at the same time. Their comprehensive knowledge, intelligence, dedication, and personal integrity helped me set a higher standard for myself on academic, professorial, and personal levels. Prof. Ulrich Flemming has been wonderfully generous with me. He often presented me with extremely insightful observations and challenges that helped me during all stages of my research. Prof. Christos Faloutsos has provided much valuable advice that has enhanced this work. Prof. Wilpen Gorr was especially helping in improving and shaping my research objectives. Prof. Ömer Akin has guided me by his valuable comments and critical remarks.

I have been very fortunate to be a member of a very talented group of students during my stay at CMU and would like to especially mention Safwan Ali, Mohammad Mogahed, Hesham Eissa, Magd Donia, Halil Erhan, Jihyun Lee, Hoda Mustafa, and Ipek Ozkaya.

Finally, the greatest appreciation goes to my wife Nuha, my sons Hamzah and Albaraa, and my daughter Samirah. I dedicate this work to them.

Contents

Abstract	4
Acknowledgement	5
Contents	6
Figures	12
Tables	14
CHAPTER 1	15
Introduction	15
CHAPTER 2	20
Background	20
2.1 Research in Computer-Supported Urban Design	20
2.1.1 Definition of Urban Design	21
2.1.2 Computer Tools for Urban Designers	22
• “Urban Simulator”	23
• 3D City Modeling	23
• “Multi-User Urban Design (MUD)”	25
• “Polytrim: Collaborative Setting for Environmental Design”	25
• “VENUE: Virtual Environments for Urban Environments”	26
• “Sketch Planning with GIS”	27

• “Interactive Multimedia Planning Support”	27
• “Online Planning”	28
• Summary	28
2.1.3 Current Digital Representations of Urban Environments	29
• DBMS Representation	29
• CAD Representation	30
• GIS Representation	30
2.2 The City of Makkah as a Case Study	31
2.2.1 The City of Makkah	32
2.2.2 Major Urban Problems in Makkah	33
• Housing	33
• Transportation (Traffic)	33
• Utilities and Facilities	34
• Development Control	35
• Land-use	35
• Guiding the Visitors of the City	36
• Urban Heritage Recording	36
2.2.3 Summary	37
 CHAPTER 3	 38
Available Data Sources	38
3.1 Digital maps (Source: Ministry of Defense)	38
3.2 ArcLand GIS Project (Source: Municipality of Makkah)	40
3.3 Hajj Housing Information System (Source: Ministry of Hajj)	43
3.4 Accommodation GIS Project (Source: The Custodian of the Two Holy Mosques Institute for Hajj Research)	44
 CHAPTER 4	 49
Promising Technologies	49
4.1 Data Warehousing	49
4.1.1 Overview	49

4.1.2	Unified Schema for Data Warehousing	51
4.2	Object-Oriented Data Modeling	52
CHAPTER 5		58
	Research Problem and Approach	58
5.1	Research Problem	58
5.2	Approach	62
5.3	Tasks	64
5.3.1	Defining Data Needs	64
5.3.2	Unified Object-Oriented Data Model	64
5.3.3	Data Warehouse	65
	• Overall System Architecture	65
	• Example Scenario	69
5.3.4	Choosing an Efficient Software Platform	73
5.3.5	Prototype Implementation and Validation	74
CHAPTER 6		75
	Urban Design Applications and Their Data Needs	75
6.1	Urban Design Applications	75
6.1.1	Visualization	76
6.1.2	Analysis	77
6.1.3	Simulation	78
6.1.4	Decision Support	79
6.1.5	Collaboration	79
6.2	Data Needed	80
6.2.1	Zones and Other Bounded Areas	81
6.2.2	Blocks	81
6.2.3	Land Parcels (Lots)	81
6.2.4	Buildings	82
6.2.5	Movement Network (Vehicular and Pedestrian)	85

6.2.6	Open Spaces	86
6.2.7	Vegetation and Landscape Features	87
6.2.8	Topography (Terrain)	88
6.2.9	Hydrography (Water System)	89
6.2.10	Facilities/Services	89
6.2.11	Utilities	90
6.2.12	Moving objects	91
CHAPTER 7		96
	Unified Object-Oriented Data Model	96
7.1	Object Models or Schemas	96
7.2	Schema Requirements	99
7.3	Urban Object Models	101
7.3.1	Geometry Classes	102
7.3.2	Zone	105
7.3.3	Block	105
7.3.4	LandParcel	106
7.3.5	Building	107
7.3.6	NetworkElement	109
7.3.7	OpenSpace	111
7.3.8	Landscape	111
7.3.9	Topography	113
7.3.10	Hydrography	113
7.3.11	Facility	115
7.3.12	Utility	117
7.3.13	MovingObject	118
CHAPTER 8		120
	Data Warehouse Components	120
8.1	Data Access Objects (DAOs)	120
8.2	Extractors	122

8.3	Solvers	124
CHAPTER 9		127
	Prototype Implementation and Validation	127
9.1	Assumptions	127
9.2	Software Platform	129
9.3	Simulating Heterogeneous Data Sources	130
9.3.1	Relational Database Management System	130
9.3.2	Computer-Aided Design System	132
9.3.3	Geographic Information System	134
9.4	Unified Object-Oriented Data Model	136
9.5	Data Access Objects	138
9.6	Extractors	140
9.6.1	DBMS Extractor	140
9.6.2	CAD Extractor	141
9.6.3	GIS Extractor	142
9.7	Solvers	143
9.8	Client Interfaces	144
9.8.1	Web Interface	145
9.8.2	Application Interfaces	149
9.9	Heterogeneous Applications	149
9.9.1	Housing Stock Reporter	150
9.9.2	3D Modeler and Visualizer	151
9.9.3	Map Generator	154
9.10	Results	155
CHAPTER 10		157
	Conclusions	157
10.1	Contributions	157

Contents

10.1.1	Solution to a Very Pressing Problem in Computer-Supported Urban Design	157
10.1.2	Urban Data Warehouse and Object Model	158
10.1.3	Portable Model, Architecture, and Implementation	160
10.1.4	Supporting Collaboration in Urban Design	160
10.2	Future Research Issues	161
10.2.1	Geometry Representation	161
10.2.2	Connected Community	162
10.2.3	Decision Support System	163
	Bibliography	165
	Appendix	170
	UML Notation	170

Figures

FIGURE 1.	The process of urban design	22
FIGURE 2.	The Key map of all the CAD maps available for Makkah (source: Ministry of Defense)	39
FIGURE 3.	Sample AutoCAD map of an urban area in Makkah (source: Ministry of Defense)	41
FIGURE 4.	ArcLand GIS system (source: Municipality of Makkah)	42
FIGURE 5.	Hajj Housing Information System (source: Ministry of Hajj)	44
FIGURE 6.	Accommodation GIS map (source: The Custodian of the Two Holy Mosques Institute for Hajj Research)	46
FIGURE 7.	Accommodation GIS, showing comprehensive information about a hotel (source: The Custodian of the Two Holy Mosques Institute for Hajj Research)	47
FIGURE 8.	Data-warehouse architecture	51
FIGURE 9.	Problems with the current process of data collection for an application	60
FIGURE 10.	Persistent and Virtual data warehousing	66
FIGURE 11.	The overall architecture of an urban data warehouse	67
FIGURE 12.	Example scenario of how the proposed warehouse works	70
FIGURE 13.	Example XML data retrieved from the data warehouse	72
FIGURE 14.	Building data	84
FIGURE 15.	Shape2D Class	103
FIGURE 16.	Solid Class	104
FIGURE 17.	Zone Class	105
FIGURE 18.	Block Class	105
FIGURE 19.	LandParcel Class	106
FIGURE 20.	Building Class	108
FIGURE 21.	Geometry of the Holy Mosque	108
FIGURE 22.	NetworkElement Class	110

FIGURE 23.	OpenSpace Class	111
FIGURE 24.	Landscape Class	112
FIGURE 25.	Topography Class	114
FIGURE 26.	Hydrography Class	115
FIGURE 27.	Facility Class	116
FIGURE 28.	Utility Class	117
FIGURE 29.	MovingObject Class	118
FIGURE 30.	Interactions between data sources, Extractors, and DAOs	122
FIGURE 31.	An Extractor has to be implemented for each kind of data source	123
FIGURE 32.	Solvers in the data warehouse	126
FIGURE 33.	Simulated DBMS (MS Access) data source with sample data	132
FIGURE 34.	Simulated CAD (AutoCAD) data source with sample data	134
FIGURE 35.	Simulated GIS (ArcView) data source with sample data	136
FIGURE 36.	Building class definition in Java code	137
FIGURE 37.	BuildingDAO class definition in Java code	139
FIGURE 38.	Different key attributes (IDs) problem	143
FIGURE 39.	Lookup table used by the Solver	144
FIGURE 40.	Web Interface to the Urban Data Warehouse, showing a list of urban object types	145
FIGURE 41.	Web Interface, showing a list of available attributes for a Building object	146
FIGURE 42.	Sample extracted data represented as HTML Tables	147
FIGURE 43.	The content of the XML file that includes the sample extracted data	148
FIGURE 44.	Housing stock reports generated using the imported XML data	151
FIGURE 45.	3D Visualization of building data extracted from XML data	153
FIGURE 46.	Maps generated from XML data	154

Tables

Table 1.	Heterogeneity of currently available urban data in Makkah	48
Table 2.	Movement network elements	86
Table 3.	Data needs for different urban design applications (Part I)	94
Table 4.	Data needs for different urban design applications (Part 2)	95

CHAPTER 1

Introduction

One of the most important concerns in rapidly growing cities is the need for careful and effective urban design. This need is especially acute in the city of Makkah, Saudi Arabia. More than three million people visit the city every year on a pilgrimage (Hajj), and, with improvements in mass transportation, this number increases every year. Accommodating millions of visitors every year and providing them with adequate utilities, facilities, and services is an enormous challenge, as is managing the movement of vehicles and pedestrians during Hajj. Local authorities are also faced with the important task of providing the local population with adequate housing.

In their decision-making, urban designers or planners rely on data provided by various sources in Makkah. Currently, these data sources are heterogeneous because they are maintained by different authorities at different locations and use different platforms, file formats, and

schemas. This becomes apparent when looking at the major data sources currently available in Makkah:

- The ArcLand GIS project at the Municipality of Makkah is a GIS system that contains land parcel information and survey maps and uses ArcView GIS software.
- The Hajj Housing Information System at The Ministry of Hajj is a relational database management system (DBMS) that provides comprehensive information about the housing units available to pilgrims and uses Oracle software.
- The Saudi Military provides digital aerial maps in the form of AutoCAD files. These files comprise various layers that capture geometric data on land parcels, streets, contour lines, elevation points, buildings, and other urban objects.
- The Accommodation GIS project at The Custodian of the Two Holy Mosques Institute for Hajj Research is a GIS system that provides comprehensive information (including location maps, pictures, and non-geometric data) about hotels and furnished-apartment buildings available for visitors to the city.

If urban designers and planners want to use software applications that need input from these heterogeneous sources, they must manually transform these data into the required format, a time-consuming and

error-prone task that may discourage their use of the software at all, thus depriving designers of possibly crucial decision-support tools.

The situation in Makkah is not unique. Urban designers and planners are faced with similar problems throughout the world.

Urban design software typically needs data in the following categories: zoning, administrative and political boundaries, topography, land parcels, buildings, monuments and historic landmarks, open spaces, traffic networks, facilities and amenities, water system, landscaping, accessibility, and utilities, including water, gas, electricity, and telecommunication. The same urban data may be required by different applications, and a single application may require data in multiple categories and from multiple sources.

This dissertation shows that a data warehouse, originally conceived for business applications, can also be used as a layer between urban data sources and urban design applications that hides the heterogeneity of the sources from individual applications. At the same time, the urban data warehouse takes the many-to-many relations between heterogeneous data sources and heterogeneous applications into account by making the data available to all applications through a uniform interface. This interface is based on a unified object-oriented data model that integrates the data independently of their source.

The research described here has been sponsored by The Custodian of the Two Holy Mosques Institute for Hajj Research at Umm Al-Qura University, Makkah. The Institute provides consultative services to

various local authorities in Makkah in order to facilitate the Hajj. One of the main goals of the Institute is to establish a data bank about Hajj in order to provide a comprehensive scientific reference that can support research and development. The following organizations have also supported this study:

- The GIS Center at the Municipality of Makkah
- The GIS and Remote Sensing Unit at Umm Al-Qura University
- The Ministry of Hajj

This dissertation is organized as follows. Chapter 2 provides background information on two topics: The first one is a literature review of computer-supported urban design. The second one introduces the city of Makkah as a case study. Chapter 3 presents the currently available sources of urban data for Makkah. Chapter 4 introduces two relevant technologies for the context of this dissertation: data warehousing and object-oriented data modeling. Chapter 5 states the problem addressed in this research and introduces the approach that has been followed to address this problem. Chapter 6 explores the various possible urban design applications and the data needed to support these applications. Chapter 7 develops a unified object-oriented data model able to capture all of the data identified. Chapter 8 describes in detail three major components of the proposed urban data warehouse: Data Access Objects, Extractors, and Solvers. Chapter 9 presents a first prototype implementation of the data model and the data warehouse, which is used to validate the object model and warehouse architecture. Chapter 10 states contributions and

conclusions of this research and discusses possible issues for future research.

CHAPTER 2

Background

This chapter provides background information on two issues: the literature on computer-supported urban design, and the city of Makkah as a case study.

2.1 Research in Computer-Supported Urban Design

This section presents an up to date review of research conducted on computer-supported urban design, starting with a brief definition of urban design.

2.1.1 Definition of Urban Design

This dissertation aims to support urban design. This section gives a brief introduction to the field of urban design, including its scale, focus, and process. The field of architecture is usually focused on buildings and their sites. By contrast, the field of planning is concerned with a larger scale, concentrating on the policies and programs that affect an entire neighborhood, city, or region. Urban design spans these two fields by combining an interest in physical design with urban policy making.

Here are some of the definitions of urban design by leading specialists. Kevin Lynch states that “Urban design deals with the form of possible urban environments.” [Lynch 1990]. Jonathan Barnett defines urban design as “the process of giving physical design direction to urban growth, conservation, and change.” [Barnett 1982]. According to Michel Batty, urban design “sits at the interface between architecture and planning, and its emphasis on physical attributes usually restricts its scale of operation to arrangements of streets, buildings, and landscapes.” [Batty et al. 1998].

To investigate data needs for urban design applications, it is important to understand the process of urban design. Urban design includes various phases (Figure 1): collecting and analyzing urban data, formalizing goals and objectives, generating and evaluating possible solutions, and translating solutions into policies and programs [Shirvani 1985]. As shown in Figure 1, data collection is the first phase in the

process of urban design. This current dissertation intends to support specifically this phase. The term "supported" in the title of this dissertation should be understood in this restricted sense.

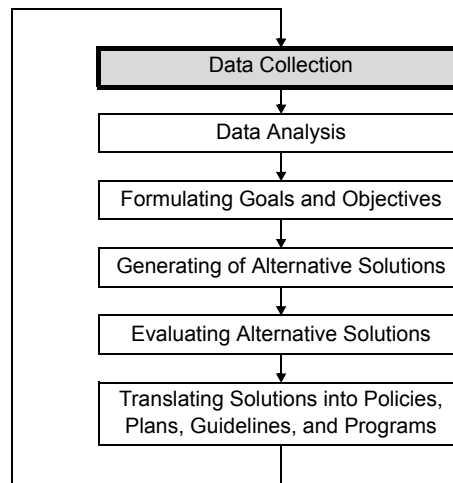


FIGURE 1. The process of urban design

2.1.2 Computer Tools for Urban Designers

This section describes the main technologies to support urban design under investigation at various research institutions, including Computer Aided Design (CAD), and Geographic Information Systems (GIS), Multimedia, Virtual Reality (VR), and the Internet. The following are some of the recent research projects that have employed/developed these technologies for urban applications.

2.1.2.1 “Urban Simulator”

This research project has been conducted at the School of Architecture and Urban Design, University of California Los Angeles. The Urban Simulator software integrates existing systems like CAD and GIS with real-time visual simulation. The purpose of the system is to facilitate the modeling, display, and evaluation of alternative proposed urban environments. The system can be used to visualize urban areas as they currently exist or to visualize proposed urban design solutions. Using this system, the project team is creating a real-time virtual model of the entire city of Los Angeles. The model is being constructed by combining aerial photographs with street level imagery and 3D geometry. This model is used for realistic 3D visual simulation of dense urban environments within Los Angeles. The system is being extended to support a client-server capability to allow a seamless interactive navigation of the entire model, which enables simultaneous interactions with the model by hundreds of remote users [Liggett et al. 1995] & [Liggett & Jepson 1995].

2.1.2.2 3D City Modeling

3D city modeling is another important research line in computer-supported urban design. Here, three examples of 3D city models are presented. The Center for Advanced Studies in Architecture (CASA) at the University of Bath has constructed a 3D computer model of the city of Bath. The model was originally built to support the formal urban planning process for the city and used by city planners to test the visual

impact of the various proposed developments on the city of Bath. The model was constructed using CAD software; it includes about 160 sub-blocks. Each block is modeled in four levels of detail. Level 1 includes a simple volumetric description of each terrace with a flat roof. At level 2 each building is modeled with accurate roof and wall geometry and tagged as a separate object in the model. Doors, windows, parapets, party walls, and freestanding garden walls are added in level 3. Level 4 shows architectural details such as chimney pots, string courses, and pilasters. At level 4 some photographic textures maps are also included for windows and shop-fronts. Furthermore, a system is built to translate the CAD model to a VRML model. The VRML model runs within an Internet browser. This allows a collaborative use of the model, where individuals and business entities may participate in adding more information to the model [Day 1994], [Day & Radford 1998] & [Day et al. 1996].

The model of the city of Glasgow created at ABACUS (Architecture & Building Aids Computer Unit Strathclyde) at the University of Strathclyde is another example of this kind of application. A three-dimensional model of the city has been constructed and is used for indexing building information and land use in order to perform quantitative and spatial analysis [Day 1994].

The last example of CAD urban models of cities is the 3D block model of the city of Adelaide. The model captures buildings, roads, infrastructure, and some zoning information [Day 1998].

2.1.2.3 “Multi-User Urban Design (MUD)”

This project has been developed at the Sundance Lab for Computing in Design and Planning at the University of Colorado in Denver. In the view of the MUD developers, urban design involves multiple stakeholders with various agendas. Each agenda forms a set of criteria, and each criterion is a single goal or property of design. The MUD program aims to promote understanding and negotiation among these stakeholders. The system enables stakeholders with various backgrounds and values to exchange design proposals and agendas. This helps to advance understanding and simulate negotiations, providing a collaborative environment for judging alternative design solutions [Gross et al. 1997].

2.1.2.4 “Polytrim: Collaborative Setting for Environmental Design”

Polytrim is a test-bed software system intended to facilitate research, teaching, and practice of landscape architecture. This project is conducted at the Center for Landscape Research (CLR) at the University of Toronto. The software aims to integrate the various formats representing landscape into one computational system. The targeted users of the system are architects, urban designers, planners, and environmental scientists. The system combines various kinds of data and computational tools found in a range of digital design and planning media, including Virtual Reality, GIS, CAD, animation, sound, hyper-links, collaborative-work via Internet, image processing, text, digital library tools, database, interactive exhibitions, customizable

interfaces, user history tracking, scripting language, and “C” library support [Danahy & Hoinkes 1995].

2.1.2.5 “VENUE: Virtual Environments for Urban Environments”

The VENUE research project has been undertaken by researchers at the Center for Advanced Spatial Analysis (CASA) at the University College of London. These researchers are interested in pushing GIS into urban design. GIS has mainly been used by planners. This project aims to extend a current GIS software package (ArcView) to support urban design by adding new functionality (operations). Currently, the research project focuses on the following themes:

- Adding 3D visualization capabilities to GIS: Rapid sketching and visualizing design ideas is an important task in urban design. The VENUE project team is exploring ways to link GIS and three-dimensional visualization tools by adding a functionality to GIS to produce a block model of built features in VRML format. Also, editing in 3D scenes has been implemented. This allows the user to add, remove, or modify features in a 3D view, by establishing a seamless link between 2D and 3D GIS. Another added functionality allows the conversion of 2D plan sketches into 3D form, using the VRML for visualization applications. The user can then interact with the 3D model by walking or flying around and through it.

- Adding Space Syntax to GIS: An extension (called Axwoman) to ArcView has been implemented to understand the morphological aspects of urban environments. The theory used for this is known as Space Syntax. The extension provides this functionality using a graph-based technique that offers useful tools for urban planning professionals. Space Syntax as an analytical tool allows the system to perform morphological and configurational analysis of urban environments [Batty et al. 1998].

2.1.2.6 “Sketch Planning with GIS”

This research has been conducted at the Planning Support Systems Group at the Massachusetts Institute of Technology. The project shows how GIS can be used for sketch planning. Sketch planning is a diagrammatic map-making technique used by urban planners, usually to describe key attributes of urban areas. For instance, the system can be used to generate a sketch plan showing nodes (a key concept used by Kevin Lynch to make sense of a place) in Boston. This application allows urban designers to combine their knowledge with analytical features of GIS to investigate patterns within the city [Singh 1996].

2.1.2.7 “Interactive Multimedia Planning Support”

This project has also been undertaken at the Planning Support Systems Group at the Massachusetts Institute of Technology. It uses multimedia interfaces to represent sound, image, and motion in order to facilitate

the understanding of complex urban information. Multimedia supports linking, managing, and browsing related information via cross-references. The system has developed global networking through the Internet and provides a collaborative urban planning system [Shiffer 1995].

2.1.2.8 “Online Planning”

The Online Planning project has been developed at the Center for Advanced Spatial Analysis (CASA) at the University College of London in order to provide a planning system via the Internet. The project covers issues ranging from the impact of the Internet on the planning process to techniques of improving tools for a web-based planning environment. Currently, the project includes various demonstrations of Internet-based and Virtual Reality applications for the planning and urban design community. The project is continuing to develop and explore techniques required to place the planning system on-line [Smith and Dodge 1997].

2.1.2.9 Summary

To conclude, various current research projects share the goal of providing better tools for urban designers. These projects follow the following main directions:

- adding functionality to GIS to support urban design,
- integrating CAD and GIS,
- adding 3D capabilities for urban visualization,

- employing interactive Multimedia for urban design support,
- making urban design available online (via the Internet), and
- developing collaborative urban design systems.

None of them addresses the general data integration problem introduced in Chapter 1. The following section discusses different computational data models that have been used to represent urban environments, including DBMS, CAD, and GIS.

2.1.3 Current Digital Representations of Urban Environments

Urban environments have been represented digitally in various formats. The main current data models for representing urban environments include DBMS, CAD, and GIS representations. This section describes these models and the problems involved with them in representing urban environments.

2.1.3.1 DBMS Representation

Traditional relational Database Management Systems (DBMS) have been used to represent urban areas for many years. Some cities are still using this kind of representation, which captures only non-geometric information about certain objects within an urban area. An example of this kind of representation would be a database of buildings. Such a database may include a table of all the buildings in an urban area with associated data such as name, use, owner, year built, and

condition. In contrast with CAD and GIS representations, the non-geometric representation does not capture the geometry of objects in an urban environment.

2.1.3.2 CAD Representation

A Computer Aided Design (CAD) model captures the two-dimensional and the three-dimensional geometry of various entities in an urban environment. CAD is mainly used for visualization purposes. It has been widely used for drawing 2D maps of urban areas and constructing 3D urban models. The level of detail in these 2D maps and 3D models is determined by their use. They can be used to model a whole city or a single building with detailed elevations. Some examples of CAD urban models are the city of Glasgow model project at the University of Strathclyde discussed above, the 3D block model of the city of Adelaide, and the city of Bath model [Day 1998]. In contrast with GIS, CAD captures geometric details. For example, the details of building facades can be captured by using a texture-mapping technique in which pictures of building facades are taken and then mapped on a 3D geometric model, thereby allowing colors and textures of buildings to be represented.

2.1.3.3 GIS Representation

GIS mainly supports the representation of places of a larger scale, like countries, regions, or urban areas. A major difference between CAD and GIS is that the geometries in GIS are geographically referenced.

This means that the locations of points are referenced by latitude and longitude using plan x and y coordinates. Another important feature of GIS is that it offers the possibility of linking non-geometric attributes with geometries. Furthermore, GIS provides powerful spatial analytical and visualization tools. For example, GIS can be used to calculate the shortest path between two points in an urban area or to find a building via its street address. Recently, 3D visualization capabilities have been added to GIS, allowing for the generation of 3D representations from 2D digital maps. This feature is currently used to represent urban objects as masses (for example, a building can be represented as a box). Despite these advantages, GIS provides less 3D modeling and visualization capabilities than CAD.

These are the main current representational models of urban environments. Section 4.2 introduces the object-oriented (OO) data modeling approach that is used in this research. It also compares the OO representational model with other models, including geometry-centered CAD and GIS.

2.2 The City of Makkah as a Case Study

This section introduces the city of Makkah as a case study and discusses the major urban problems facing the city.

2.2.1 The City of Makkah

Urban design is one of the most important concerns in rapidly growing cities. It is an especially important issue in the city of Makkah, Saudi Arabia. This city is considered the heart of the Islamic world, and every Muslim hopes to visit the city at least once in his or her lifetime. Nowadays, more than three million people come to the city every year on a pilgrimage (Hajj), and the number of visitors is increasing every year with improvements in mass transportation. This massive yearly influx of visitors from all over the world is dramatically changing the urban fabric of the city, especially the central area where the Holy Mosque is located. Many other problems have emerged as the city has developed, such as those caused by the location of the Mosque in a valley surrounded by mountains. The diverse cultures of the people visiting the city every year further complicate the problems. Traffic and the increasing need for housing and services also compound the problems that need to be solved. Finally, the city of Makkah has witnessed rapid development in recent decades, and many historic buildings from the Middle Ages through the 19th century have been destroyed to make room for modern development.

In short, the city of Makkah offers a fascinating context in which to examine the data model developed through this research project.

2.2.2 Major Urban Problems in Makkah

Local authorities and urban designers in Makkah are concerned with the city's major urban problems, which include housing, transportation, traffic movement, facilities, services, and urban form. A brief description of these problems follows.

2.2.2.1 Housing

Providing accommodations for more than three million people coming to the city of Makkah each year is an enormous challenge. It is a major responsibility of local authorities. The Ministry of Hajj, for instance, has established a huge database that includes information about all accommodations available for pilgrims. This systems facilitates managing, monitoring, and improving the housing of pilgrims.

Major housing concerns include the problems of the high vacancy rate around the Holy Mosque, the condition of the different types of residential buildings, the annual and Hajj rent values, the effect of Hajj on housing and building design, and the condition of the city's infrastructure as effected by Hajj [Hariri 1986].

2.2.2.2 Transportation (Traffic)

The nature of today's Hajj requires substantial planning and effort to provide support and infrastructure [Al-Yafi 1993]. An important problem facing the city of Makkah is managing the movement of vehicles and mass transit during Hajj. The huge number of people and the nature of

their activities in central Makkah further complicate the problem. Local authorities have worked on various solutions to resolve these problems. For example, small cars are not allowed to enter the central area of Makkah during peak periods. Although some solutions have been implemented, with the increasing number of pilgrims more problems arise.

Movement of pedestrians in central Makkah is a major concern for city planners and designers because of the following reasons:

- All pilgrims want to go to the Holy Mosque at the same time.
- At peak times, the Mosque can host 1.5 million people.
- Automobile and pedestrian movement are not well separated.
- Throughout the city, various activities take place other than going to the Mosque, for example shopping.

These characteristics of pilgrims' and visitors' movement in Makkah create many problems, some of them quite serious. One such problem is the crisis that can occur when the crowd panics and pilgrims are trampled to death. Many solutions have been implemented to prevent such problems, though more are still required.

2.2.2.3 Utilities and Facilities

Facilities and services are essential resources of an urban environment. The success of an urban environment depends on the

availability, accessibility, affordability, and presentability of facilities and services [Barhamain 1997].

Providing the huge number of visitors to the city with adequate utilities and facilities is a major concern for local authorities. Water, electricity, and telephone utilities have to be provided for all accommodation units available to pilgrims. Demand of many pilgrims makes this a big challenge. Parking, shopping, food, transportation, and other important facilities must also be provided.

2.2.2.4 Development Control

The urban form of the central area of Makkah is currently changing rapidly to accommodate more visitors to the city. Recently, high-rise and massive buildings have been constructed in that area. Such development is dramatically changing the skyline of central Makkah. Previously, the Holy Mosque in the central area was surrounded by a homogenous traditional urban fabric of smaller low-rise buildings. Visualizing and understanding possible future development of the city are crucial for evaluating urban design solutions because the rapid development happening in central Makkah should be under some control to provide a better city image.

2.2.2.5 Land-use

Finding efficient land-use policies is a challenge for city planners in Makkah due to the complex activities happening in the central area with the increasing number of pilgrims. Land-use patterns affect other

activities like transportation operations, which makes them important means for improving other systems.

2.2.2.6 Guiding the Visitors of the City

Because pilgrims come from many diverse cultures helping them finding their way around Makkah is a special challenge. As with tourism, visitors should be able to find what they are looking for as quickly and easily as possible. Planners in the city are concerned with providing an urban environment that helps pilgrims navigate the city.

2.2.2.7 Urban Heritage Recording

Another important problem in Makkah is how to preserve the existing urban heritage or record historic urban environments that will no longer exist. Due to the rapid development now underway in central Makkah, many valuable historic urban areas are being demolished to make room for new projects. Though they will disappear, these areas can be recorded digitally for future generations. Computers provide easier and faster information storing and retrieving capabilities than traditional heritage recording tools. For instance, a 3D model of an urban area can provide a digital database of the urban physical environment, and each entity in the urban space can be linked with non-geometric historic information [Koshak & Gross 1998].

2.2.3 Summary

To conclude, Makkah is a rapidly growing city that has a particularly acute need for effective and sustainable urban design. Accommodating millions of visitors every year and providing them with appropriate housing as well as adequate utilities, facilities, and services is an enormous challenge, as is facilitating the movement of vehicles and pedestrians during Hajj.

Decision makers are relying increasingly on software applications to support their decisions, but the use of these applications is severely restricted by the heterogeneity of the data sources on which they have to rely. These sources are described in greater detail in the next chapter.

CHAPTER 3

Available Data Sources

This chapter presents the currently available sources of urban data in Makkah to support urban design. These data can be used as input for the data model developed through this research.

3.1 Digital maps (Source: Ministry of Defense)

The digital maps available for the city of Makkah are generated from aerial photographs taken by the Saudi Ministry of Defense. These maps are currently available in AutoCAD's DXF file format for use by many local authorities in Makkah. These maps were created in 1989. The city has dramatically changed since then, especially the central area of

Makkah, where the huge expansion of the Holy Mosque has changed the urban pattern around the mosque.

The GIS Center at the Municipality of Makkah has been working on updating these maps. The Center has also added more information to these maps, like zoning. Figure 2 shows a key map, which includes all the maps available for the city. The key map can be used to select the area for which a more detailed map should be retrieved.

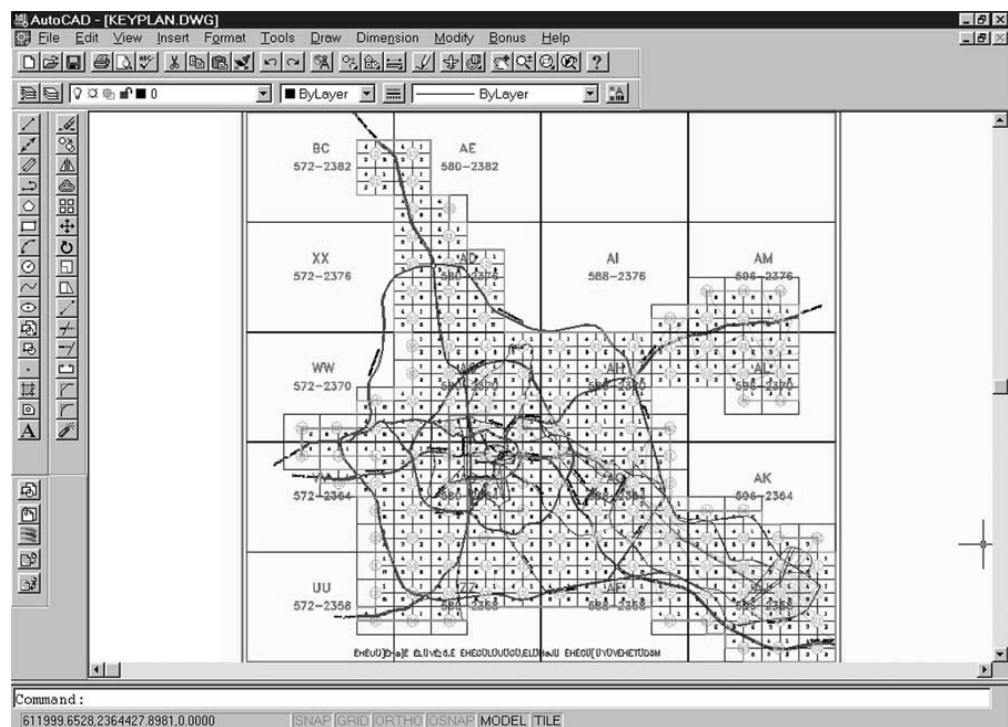


FIGURE 2. The Key map of all the CAD maps available for Makkah (source: Ministry of Defense)

Each map (Figure 3) is available in the AutoCAD's DXF file format and includes layers that represent the following urban objects or elements:

- zoning,
- main beltway (ring) roads,
- streets and paths,
- walkways,
- land parcels,
- blocks,
- buildings,
- contour lines,
- elevation points,
- street trees and lights,
- street names,
- mosques and school locations, and
- landscaping elements.

3.2 ArcLand GIS Project (Source: Municipality of Makkah)

- ArcLand is an extension of the ArcView GIS software that supports managing land parcel information and survey maps. It was developed at the GIS Center at the Municipality of Makkah. ArcLand allows users to enter, update, search for, and visualize

land parcel information. The system also serves as a useful tool in managing the issuing of building permits.

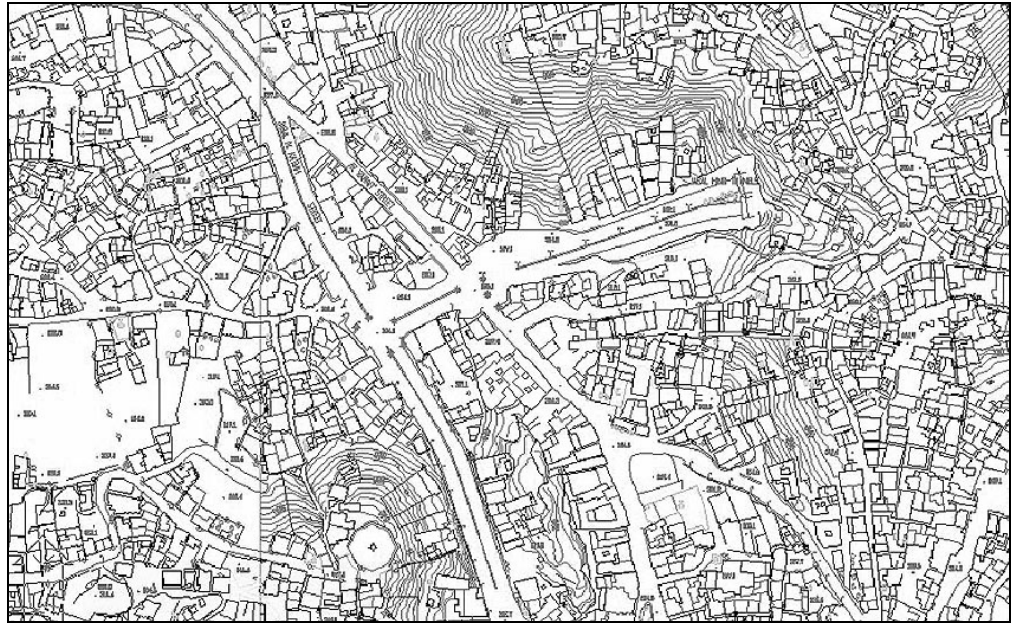


FIGURE 3. Sample AutoCAD map of an urban area in Makkah
(source: Ministry of Defense)

Figure 4 shows the interface of the ArcLand system, which can be used to enter or retrieve a survey map and non-geometric information on a particular land parcel. The system stores the following information about each land parcel:

- neighborhood name,
- municipality name,
- deed number,

- deed type,
- deed date,
- deed source,
- owner name, and
- land parcel geometry (GIS map) as a closed polygon.

إعدادات كروكي

بيانات العامة

الموضوع رخصة ما، الموقع العربية البلدية العربية

مقياس المظهر الكبير مقياس المظهر الصغير حجم الورق

بيانات المسك

رقم المسك ١٣٥٦ نوع المسك دائرة مستخدم مستخدم

تاريخ المسك ١٤١٤ ١٢ ٣٥

بيانات صاحب المدة

اسم الأول محمود اسم الأب محمد اسم الجد محمد اسم العائلة حمد

رقم الخريطة ١٣٣٤٥٦ مصدر الخريطة الرياض تاريخها ١٤١٤ ٠٢ ٠٥

رقم مدخل البيانات ١٧ وقت الإدخال ٢٠١٢ ٠٦ ٠٢ التاريخ ١٤٢٠ ٠٢ ٠٥

تدوير خصائص الكروكي

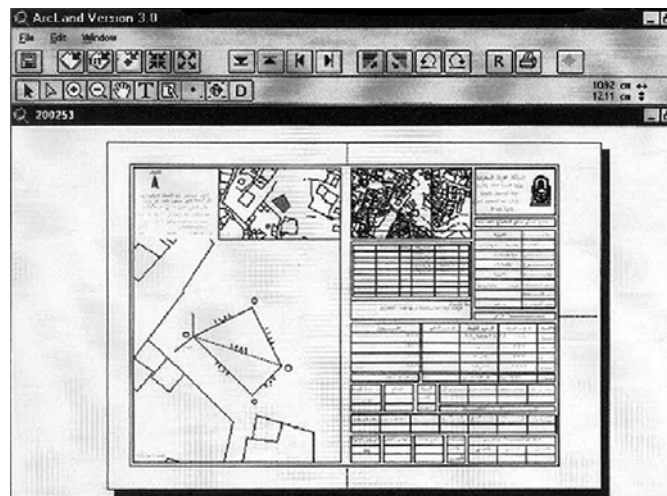


FIGURE 4. ArcLand GIS system (source: Municipality of Makkah)

3.3 Hajj Housing Information System (Source: Ministry of Hajj)

The Hajj housing information system has recently been implemented by the Ministry of Hajj. This project aims to provide comprehensive information about the housing units available for pilgrims coming to Makkah each year. Figure 5 shows the interface of the Hajj housing information system. The information available in the system includes:

- residential building ID number,
- permit number,
- building type,
- number of floors,
- number of rooms,
- owner information,
- number of pilgrims,
- occupant information,
- manager name,
- facility manager,
- lease information, and
- pilgrim data.

FIGURE 5. Hajj Housing Information System (source: Ministry of Hajj)

3.4 Accommodation GIS Project (Source: The Custodian of the Two Holy Mosques Institute for Hajj Research)

The Custodian of the Two Holy Mosques Institute for Hajj Research publishes annually a guide book on accommodations in central Makkah. This book provides detailed information about the various options available for pilgrims and visitors, which include hotels and

furnished-apartment buildings. The information about each accommodation includes:

- location map,
- address,
- contact information,
- picture of the building,
- number of rooms/apartments/suites,
- list of available building facilities, and
- list of available room services.

The Accommodation GIS Project, developed by the author, aims to convert the Accommodation Guide Book from its paper-based format to a digital one to facilitate storing, retrieving, and updating information. A relational database was created for storing the non-geometric information about each accommodation. The database has been linked with a digital map of Makkah (Figure 6), making the book available in a GIS environment. Finally, a picture of each accommodation is hyperlinked with its location on the GIS map.

A user of this system can search for a hotel either by using the non-geometric database or by navigating through the map. After selecting a hotel (Figure 7), the user can then retrieve information about each hotel:

- a table providing a non-geometric description of a hotel,
- a map showing the location of the hotel, and
- a picture of that hotel.



FIGURE 6. Accommodation GIS map (source: The Custodian of the Two Holy Mosques Institute for Hajj Research)

The GIS environment provides a link between these three pieces of information and facilitates the storing, sorting, and retrieving of the information. It also allows sophisticated analysis for research purposes on the characteristics of housing options provided for the visitors of Makkah.

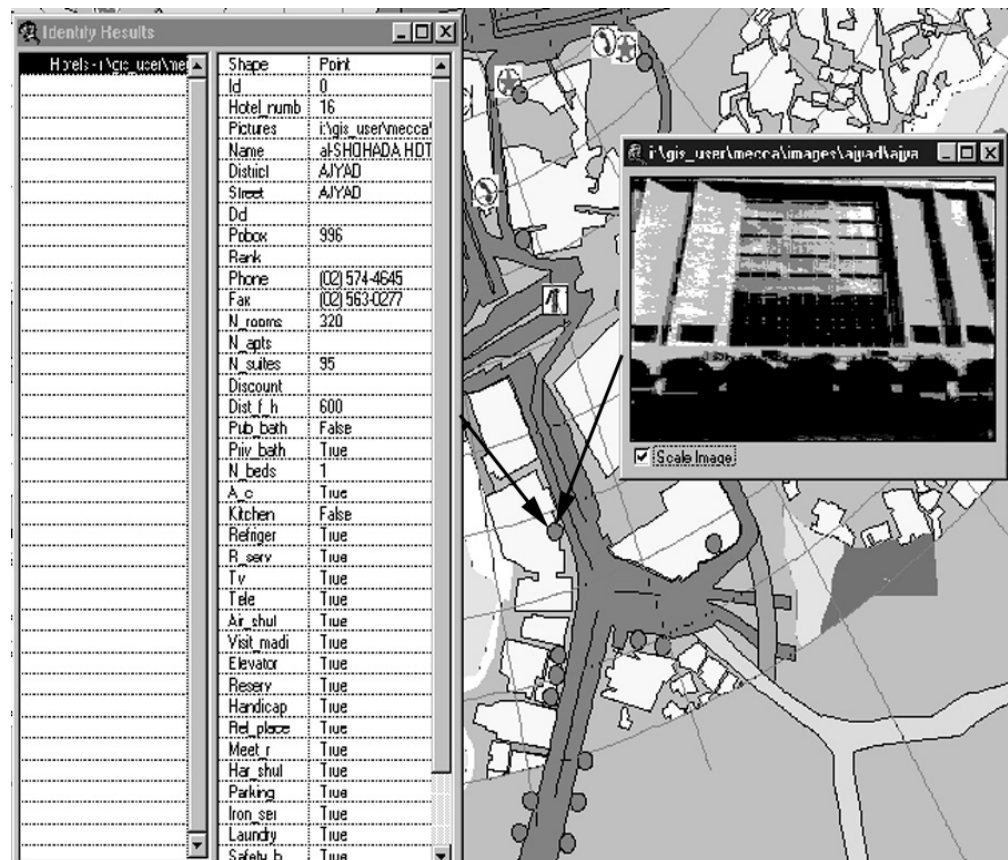


FIGURE 7. Accommodation GIS, showing comprehensive information about a hotel (source: The Custodian of the Two Holy Mosques Institute for Hajj Research)

This section has provided a brief description of the available sources of urban data in Makkah. These data can be used as an input for different urban applications. Table 1 summarizes the different platforms, different file formats, and different schemas used by these sources.

Data Source (Local Authority)	Ministry of Defense	Municipality of Makkah	Ministry of Hajj	Hajj Research Institute
System Name	Aerial Maps	ArcLand	Hajj Information System	Accommodations GIS
Purpose	national maps for all cities	land-parcel information management	pilgrims information management	accommodations data management
Platform	Unix	NT	NT	Windows
Software	AutoCAD	ArcView	Oracle	ArcView
File formats	.dwg	.dbf .shp .shx	.odb	.dbf .shp .shx
Schema and Content	collection of geometries	relational and polygons	relational	relational and points

TABLE 1. Heterogeneity of currently available urban data in Makkah

CHAPTER 4

Promising Technologies

As shown in the previous chapter, a large amount of urban data is currently available at heterogeneous data sources in cities such as Makkah. This section presents two relevant technologies that can be utilized to integrate these data and make them directly available to urban designers.

4.1 Data Warehousing

4.1.1 Overview

Large organizations or companies may have offices and facilities at many sites, where each site may collect a large volume of data. This means that different data are present at different locations on different

platforms and under different schemas. For instance, product data and customer data may be stored in separate databases at different locations. But corporate decision makers need access to information from all of these sources. Data warehousing provides a solution for this problem [Silberschatz et al. 1999].

A data warehouse (see Figure 8) is a repository (archive) of information gathered from heterogeneous data sources. The data are stored under a unified schema (model) at a single location in a separate, integrated database. The extracted data may be queried directly by a user or used as input to an application. A data warehouse thus presents clients (users or applications) with an integrated and uniform data source. Since the data sources are constructed independently and likely to have different schemas, a data warehouse must perform schema integration before data are delivered to clients. Data extraction happens periodically (like every week).

Since the data stored in a data warehouses consume a large amount of storage resources, another kind of architecture has been proposed to solve this problem. The approach is called “virtual data warehousing”. In this architecture, the data are not physically stored in the data warehouse, but extracted from their heterogeneous sources as they are requested. Figure 8 shows also the architecture of a virtual data warehouse [Hull & Zhou 1996].

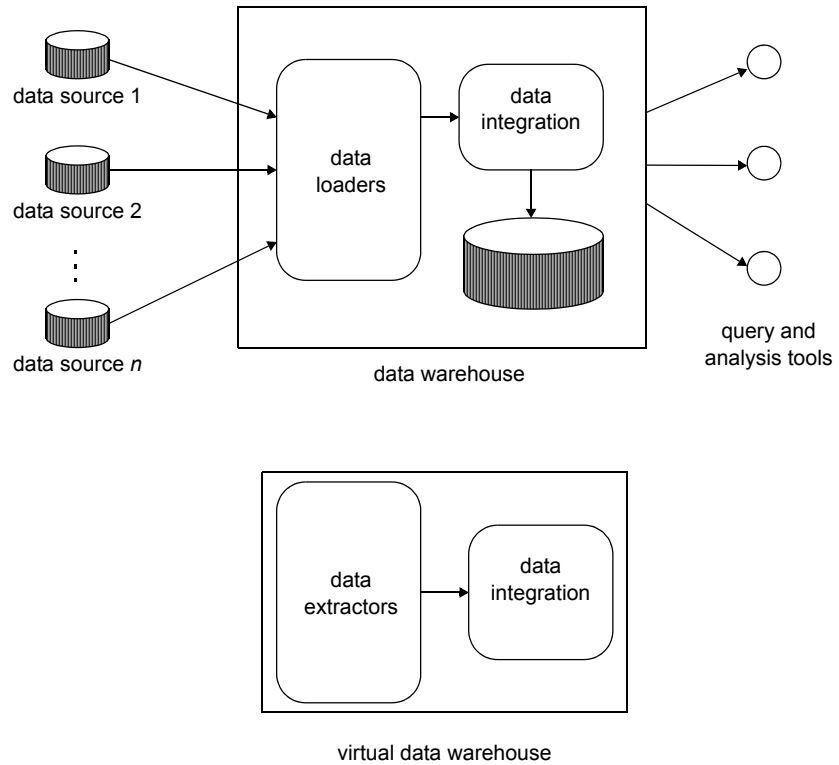


FIGURE 8. Data-warehouse architecture

4.1.2 Unified Schema for Data Warehousing

Since data sources are constructed independently, they are likely to have different schemas. A major task of a data warehouse, be it persistent or virtual, is to perform schema integration. Data coming from different sources must be converted to the integrated schema before

they are stored or delivered. This means that the data provided by the warehouse are not just a copy of the data coming from different sources. Instead, they present a uniform view of the data across all sources [Silberschatz et al. 1999].

The following section discusses alternative data models to represent urban environments. Providing a unified data model offers many advantages. First, it integrates urban data coming from different sources. Second, once the model has been designed and implemented it can be reused by various cities or regions within a country. The unified schema also facilitates generating queries over distributed and fragmented data describing a certain urban entity or an entire urban area. This helps urban data providers share the data they have with others, including urban designers. An urban designer could extract the needed data from different sources using an urban data warehouse.

4.2 Object-Oriented Data Modeling

Data warehouses require a unified data model that integrates data at the heterogeneous sources. This section discusses the benefits of an integrated object-oriented data model of urban environments vs. a geometry-based model as used in traditional CAD and GIS software.

As explained in Section 2.1.3, the conventional CAD and GIS approach of modeling the real world is geometry-centered. In this approach, data

are a collection of geometric constructs like points, lines, circles, and polygons. These constructs are the primitives in a geometry-based representation. Various constructs can be grouped by one layer or theme. For example, a building can be represented as a collection of primitives (including lines and polygons), and many buildings are grouped in one layer named “buildings”.

General drafting tools are provided in such systems to facilitate the editing of these geometries. These editing operations are independent from the kind of the real-world object the geometric elements represent. For example, the same editing operations can be performed on a polygon representing a building and another polygon representing a road. In such a data model, the behavior of a line representing a road is identical to the behavior of a line representing a stream. Using this model, urban objects are aggregated into homogeneous collections of points, lines, and polygons with generic behavior [Henderson & Ahearn 1998] & [Zeiler 1999].

In addition, non-geometric attributes may or may not be attached to these geometries. For example, building number, age, and condition information can be attached to a polygon representing the footprint of a building. The geometry files (data) and non-geometric files (data) are separated and they are not integrated [Henderson & Ahearn 1998] & [Zeiler 1999].

The Object-Oriented (OO) modeling approach relies on abstract data types. These data types are called object classes. Objects are instances of these classes and typically represent real-world objects of

interest. These objects are the basic primitives (entities) in an object-oriented representation. An object contains attributes (fields) and methods (operations). Attributes capture *all* (geometric and non-geometric) data needed to describe a particular object. For example, the attributes of a building object include the building geometry (outline, height etc.), name, address, age, and all other information related to that building. Methods (also called member functions) support operations that manipulate object data or states. For instance, a building object may contain methods like calculate footprint area; get building age; set building height; update land-use; and show roof shape. Each object captures both the attributes and the behavior (actions that manipulate attributes) of that object [Meyer 1988], [Henderson & Ahearn 1998], [Zeiler 1999] & [Sola-Morales 2000].

If one applies object-oriented data modeling in this research context, an urban environment can be represented as a collection of domain objects like buildings, parcels, and streets. Each object captures all geometric and non-geometric attributes of an urban entity. Relationships between various urban objects and how they interact can be also captured. For example, a “Building” object will contain all attributes of a specific building in one package. It is for these reasons that an object-oriented data model is particularly promising for the type of integrated schema needed by the data warehouse.

Aside from these specific advantages, object-oriented programming is becoming increasingly popular because it supports software quality factors including correctness, robustness, extendibility, reusability, and

compatibility. These external factors are perceivable by end-users. Correctness is the ability of a system to exactly perform its tasks as defined by requirements and specifications. Robustness is the ability of a system to work even in abnormal situations. Reusability is the ability of a system (part or whole) to be reused for new applications. Extendibility is the ability of a system to be adapted to changes of specifications. Compatibility is the ability of a system to be combined with others [Meyer 1988].

Internal factors that are perceivable only by software professionals are necessary to achieve these external factors. To achieve correctness and robustness, systematic development based on explicit specifications and constraints is needed. To achieve extendibility, reusability, and compatibility, a flexible and decentralized design consisting of modules with well-defined interfaces is needed [Meyer 1988].

The key to achieve these aims is modularity, which provides a flexible system architecture. Five criteria help establish design principles with respect to modularity: decomposability, composability, understandability, continuity, and protection. To fulfill these criteria, the following design principles should be applied to software development according to Meyer.

- Linguistic modular units: Modules have to correspond to syntactic units of the language used. This principle supports

decomposability, composability, understandability, and protection.

- Few interfaces between modules: Every module should communicate with others as seldom as possible. This principle follows continuity and protection.
- Small interfaces: Whenever two modules communicate, they should exchange as little information as possible. This principle follows continuity and protection.
- Explicit interfaces: Whenever two modules communicate, it should be obvious from the interface of any or both of them. This principle follows decomposability and composability.
- Information hiding: All information about a module should be private unless it is declared public. This principle follows decomposability, composability, understandability, and continuity.
- Open-closed principle: A module is considered open if it is still subject to extension, where a module is considered closed if it is available for use by other modules. This principle follows decomposability and composability [Meyer 1988].

In the object-oriented approach, systems are modularized on the basis of their data types expressed as classes and every class represents a particular data type implementation. Objects are run-time instances of a class. Classes should be designed to be as general and reusable as

possible. A class offers a number of services that can be requested by clients (users) of that class [Meyer 1988].

Classes are connected by two relations: client and descendant (inheritance). Inheritance allows a class to be designed as an extension or restriction of another class. Another important feature is polymorphism which is the ability of an entity to become attached to objects of various possible types. All these modularity features support system extendibility and reusability [Meyer 1988].

Because of these features, software developers are following more and more the object-oriented paradigm. This is another reason to make the unified schema needed by the data warehouse object-oriented: it facilitates data import by object-oriented applications.

CHAPTER 5

Research Problem and Approach

This chapter defines the research problem investigated in this dissertation, and outlines the approach followed to solve the problem.

5.1 Research Problem

The problem addressed in this dissertation confronts cities and planning authorities all over the world. Local authorities are moving towards storing and managing urban data in digital form. However, the data storage devices used are heterogeneous and typically include relational database management systems (DBMS), GIS, and CAD files. As a result, data are present in different locations on different platforms and under different schemas. This poses a problem for software

applications meant to support urban design that require input from more than one data source.

Suppose an urban designer needs to collect data about a certain urban area to be used for a particular urban design application. As shown in Figure 9, collecting the needed data means going to each data provider physically located in various places and requesting the needed data. The figure shows that heterogeneity of data can occur in three typical situations:

1. Different platforms: Local authority A uses platform P1, while local authority B uses platform P2. For example, local authority A may use MS Windows and local authority B may use the Unix platform, resulting in incompatibility of platforms.
2. Different file formats: Local authorities B and C use the same platform F2, but use two different applications and provide data in two different file formats (format F1 and format F2). For example, local authority B may use AutoCAD DXF files and local authority C may use ArcView GIS Shape files, resulting in incompatibility of file formats.
3. Different schemas: Local authorities C and D use the same platform P2 and the same file format F2, but use two different schemas (S1 and S2). For example, both local authorities C and D may use Oracle as a DBMS to manage urban data. But local authority C may use a very simple schema of one flat table, while local authority D may use a more sophisticated schema of

multiple tables and complicated cardinalities. Even if both data sources use a simple table to store data, they may use different naming conventions for the attributes. This results in incompatibility of schemas.

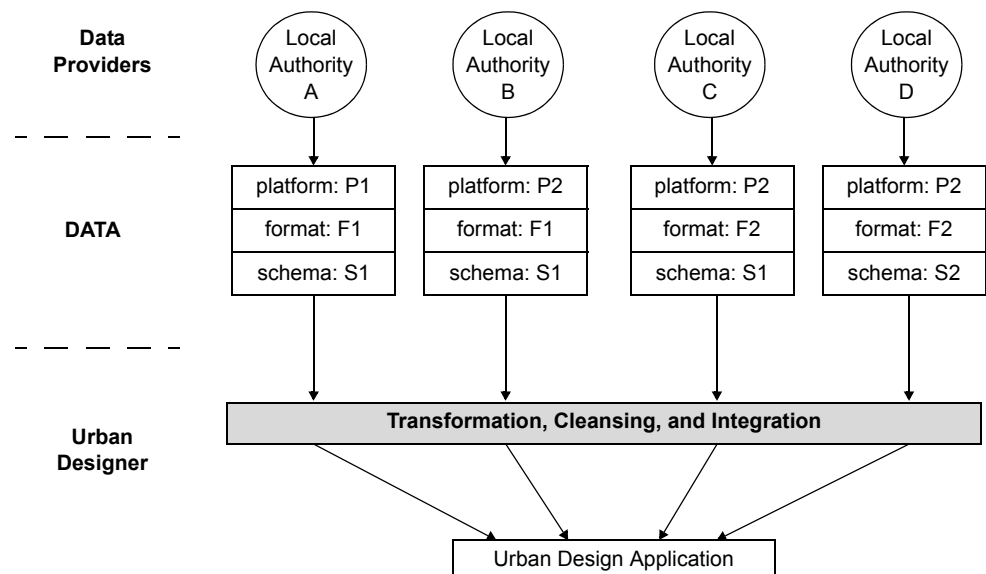


FIGURE 9. Problems with the current process of data collection for an application

The situation of incompatible platforms, file formats, and schemas requires a considerable amount of work to import the data coming from heterogeneous sources into a specific urban design application. An urban designer has to go through the lengthy and tedious task of

transforming, cleaning, and integrating the needed urban data, so that they can be used by an urban design application.

In addition, the available urban data may be used by heterogeneous applications. These applications are likely to run on different platforms, using different software, and under different schemas. This means that data collection, translation, and import have to be performed for each application an urban designer wants to use in decision-making. The time and effort required to do this may be so great that may preclude using the application at all, thus depriving urban designers of possibly crucial design support tools. To solve this problem, the needed data should be provided to urban designers in a standard file format. They should also be provided in one simple unified schema (data model). None of the recent/current research projects discussed through Section 2.1.2 addresses this problem in a comprehensive data integration approach.

This dissertation aims to provide a solution for this central problem in computer-supported urban design and planning. Its goal is to develop a mechanism that hides the heterogeneity of urban data sources and is able to provide the data available in these sources to urban designers and the applications they are using in a unified format. This mechanism has to be general and flexible enough to handle data sources on any of the commonly available platforms, using any typical software (DBMS, CAD, or GIS) and any schema appropriate for this software. Furthermore, it must be possible to implement this mechanism using commonly available, platform-independent software and tools. This

generality and flexibility is meant to assure that the mechanism can be implemented not only for the city of Makkah, but by many other cities that are faced with the same problem.

5.2 Approach

As discussed in Section 4.1, data warehousing provides an ideal solution for the problems discussed in the previous section. It is able to provide urban designers with a single unified interface to collect urban data that are physically available at heterogeneous sources. It hides the heterogeneity of the data sources from urban designers. Any local authority can join the data warehouse as a data provider. In addition, any urban designer can retrieve the needed urban data using a publicly available interface (via the Internet).

A major requirement for building a data warehouse is to develop a unified data model (schema) that integrates the data coming from heterogeneous sources. The model must be able to capture all the data needed by urban design applications and all the information describing the various urban objects available at different local authorities. As discussed in Section 4.2, object-oriented data modeling offers important features that facilitate developing such a data model.

Such a data model offers many advantages. First, it facilitates integrating the currently fragmented data about urban entities. Second,

it facilitates generating queries using a single view of data. Third, it can be implemented by different cities within a region or a country, so that the same application can receive data from different communities regardless of the data sources and formats.

In this research, the city of Makkah is used as a test case, not an end in itself. It is used to validate the hypothesis behind the data warehouse and the data model. The city is a promising test case because it clearly experiences the problem addressed above; in fact, it is especially acute for this city with its need for a sustained and integrated urban design effort. Clearly, a dissertation like the present one cannot collect the data available in heterogeneous sources in cities all over the world and test the hypothesis on such a comprehensive data set. The expectation is, rather, that an approach that works for Makkah is likely to work for many other cities facing a similar situation. For example, once the model has been validated, any city within Saudi Arabia (like Madinah, Riyadh, or Jeddah) can implement the same data model and warehouse to support that city's urban design applications.

In order to achieve the goals set for the data model and the data warehouse, the tasks described in the following sections have to be executed.

5.3 Tasks

5.3.1 Defining Data Needs

The first major task of this research is to determine the *possible urban design applications* and the *data needed* to support these applications. This task will be investigated for representative urban design and planning applications and is discussed in detail in Chapter 6.

5.3.2 Unified Object-Oriented Data Model

The second major task of this research is to develop a unified object-oriented data model to represent urban environments. The model has to capture the data identified in the previous task and enable the data warehouse to integrate the data retrieved from heterogeneous sources. This model captures all entities of interest in the form of objects belonging to classes (data types), the attributes of each object and the relationships between different objects. Chapter 7 describes the unified object model developed.

For this dissertation, the data model was designed using UML, the Unified Modeling Language [Booch et al. 1999], which has become the standard language used for modeling software application requirements and design. UML provides development teams with a common

language and allows team members to communicate with each other [Naiburg & Maksimchuk 2001].

5.3.3 Data Warehouse

The task of designing and developing the data warehouse is the third major task required by this research. This section introduces the overall architecture for the warehouse and illustrates its function through a sample scenario.

5.3.3.1 Overall System Architecture

As mentioned in Section 4.1, there are two basic options for a data warehouse implementation (Figure 10):

The *persistent data warehouse* stores all data that may have to be exported to clients in a separate, integrated, and comprehensive database. The data are collected frequently (weekly or monthly, for instance) from the heterogeneous sources and stored physically in a separate database, which is the central component of the data warehouse. Data requests by clients are sent directly to and returned by this database. This option is problematic when the amount of data coming from heterogeneous sources is large and consumes a large

amount of storage resources. In addition, data stored in the warehouse are not always up-to-date.

The *virtual data warehouse* is more attractive in the present context. In this option, data are collected from heterogeneous sources as needed (“lazy” collection). When a client requires data, the data warehouse connects remotely to the appropriate data sources and returns the requested data to the client without storing any data within the data warehouse itself.

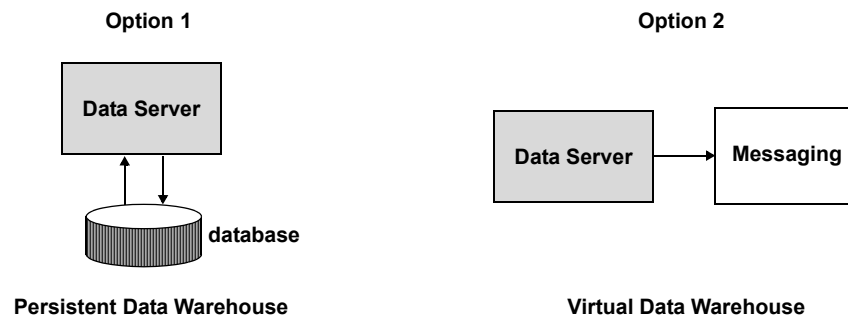


FIGURE 10. Persistent and Virtual data warehousing

The overall architecture of a virtual data warehouse to support urban design is shown in Figure 11. A client (urban designer) requests certain data for direct inspection in a web browser or as input for a specific application. The client application (or web browser) sends the data requests to *Data Access Objects* (DAOs) in the form of queries. A DAO is a software component that is able to import attribute data for a

specific class in the underlying schema from various sources and to export the data to clients in a unified form determined by the schema. The DAO concept is explained in more detail in Section 8.1.

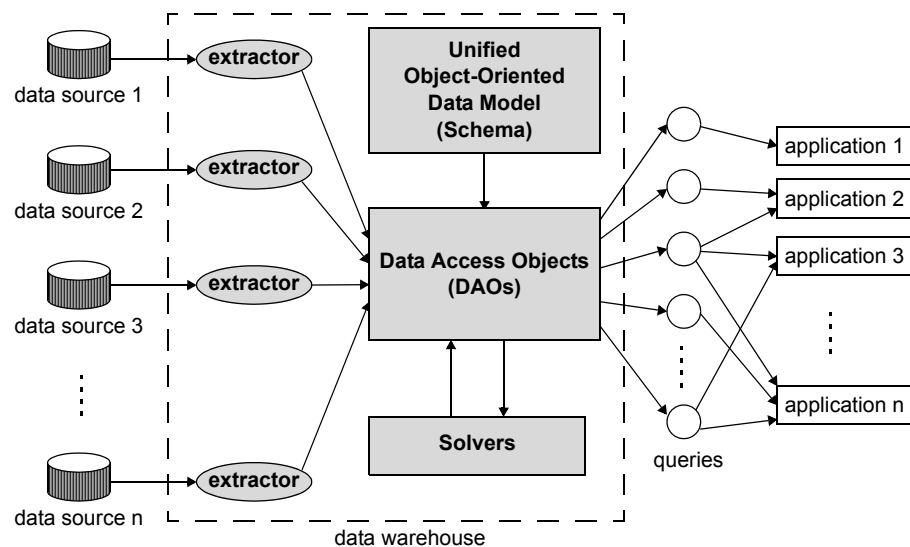


FIGURE 11. The overall architecture of an urban data warehouse

Depending on the location of data and the type of data source, the DAO uses appropriate *Extractors* to extract the needed data from heterogeneous data sources. Extractors communicate between the DAO and the various data sources, including DBMS, CAD, or GIS. An extractor needs to be implemented for each data source. It takes a query and the location of a data source as arguments from a DAO and

returns the requested data to the DAO. Extractors are explained in greater detail in Section 8.2.

The data returned from different extractors have the form of objects as defined in the underlying object model. Such an object represents a corresponding object in the urban environment and captures all the different data that describe it. This representation hides the different sources of the data. In cases where there are integration problems, solvers are used. A *solver* is a software component that handles a particular data integrating problem. Solvers and the kind of problems they handle are discussed in more detail in Section 8.3.

After the data have been integrated and unified, the warehouse returns the requested data to the user through a web browser or sends the data directly to an application. The features of this architecture are enhanced as the design, implementation, and testing of the urban data warehouse continue.

To preserve data integrity, the proposed data warehouse does not accept any data from end users (urban designers), because they could modify the state of the data sources. Only local authorities are allowed to modify persistent data. However, the process of urban design that is illustrated in Figure 1 clearly generates additional information during the various design stages. But the addition of these data to the warehouse

has to go through the local authorities managing the warehouse, who have to decide if and when new data are to be added to the warehouse.

Different levels of access are needed to provide data privacy. There are certain data that need not to be provided for public, such as the personal information about an owner of a building. For this reason, the warehouse administrator could provide two levels of access to two different groups of users. Decision makers can be provided with full access, while urban designers can have limited access. This is not included in the proposed architecture, but it could possibly and easily be incorporated by adding a software layer that provides access control. For example, a user name and password information can be requested up front. According to this information, the data warehouse can determine access privileges.

5.3.3.2 Example Scenario

An example scenario of using a virtual urban data warehouse is illustrated in Figure 12. Suppose an urban designer needs to obtain information about all of the buildings within a specific neighborhood and that, for each building, the following attributes are required:

- building identification number (ID),
- street address,
- height,
- number of occupants,
- year built, and
- use.

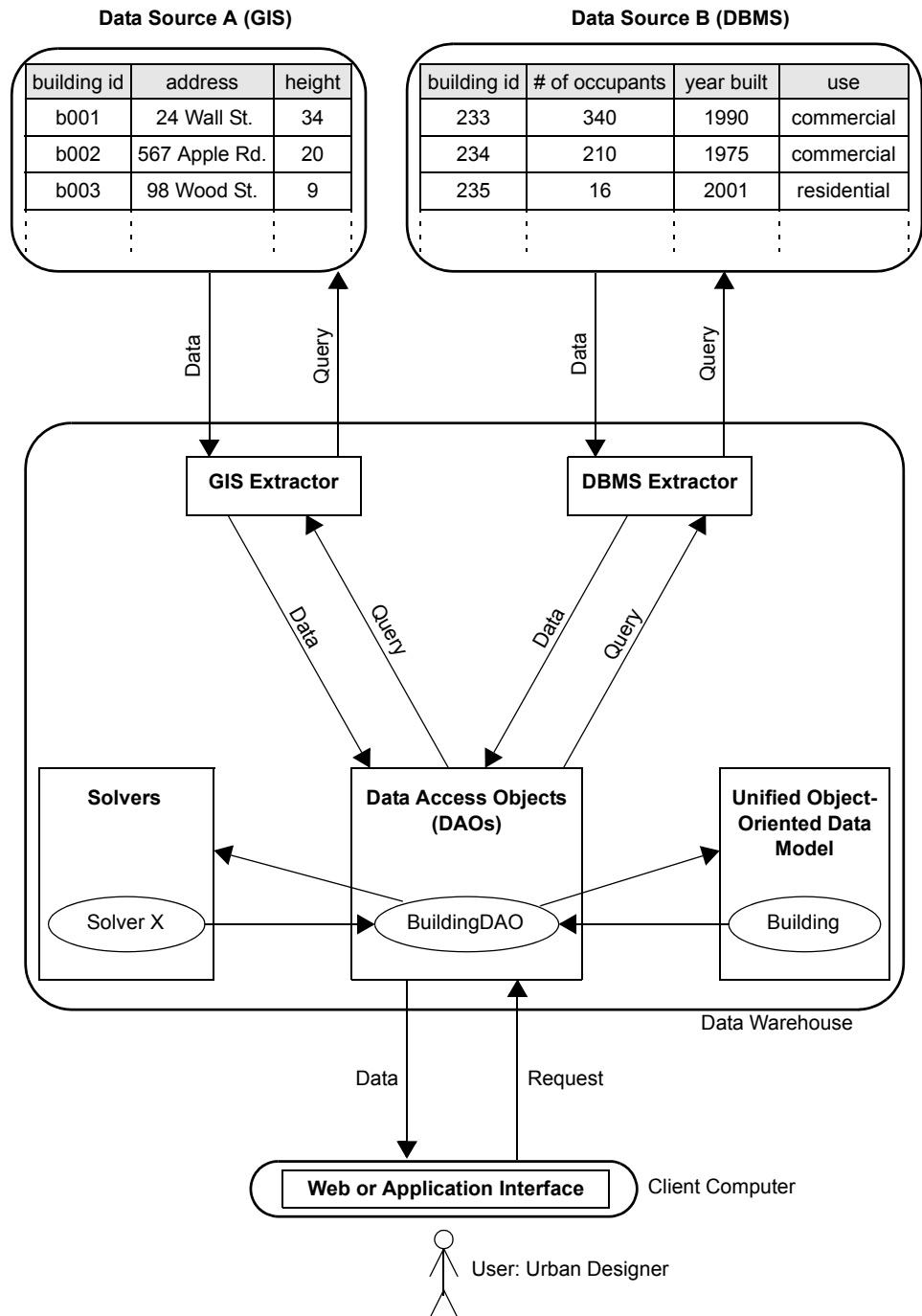


FIGURE 12. Example scenario of how the proposed warehouse works

Suppose also that the needed data are fragmented vertically in two different data sources. Building address and height information are available at data source A, which is a GIS, but the number of occupants, year built, and use information are located at source B, which is a DBMS.

Through a web browser or an API (Application Programming Interface), the urban designer identifies the needed data. The browser or the API then sends the data request to the data server that hosts the data warehouse. A Data Access Object receives the request; in this case, it would be a “BuildingDAO”. The BuildingDAO next sends the data request to the appropriate extractors. In this example, the BuildingDAO splits the data request into two queries. The first query is sent to data source A to extract the building address and height information using a GIS extractor. The second query is sent to data source B to obtain the number of occupants, year built, and use information using a DBMS extractor.

The unified data model is then used to integrate the data coming from the heterogeneous sources. In this example, a “Building” object is created to capture all data coming from data sources A and B. This Building object has attributes that represent the requested data in a unified form.

In cases where there are integration problems, solvers are used. Suppose that in this example, the key attribute at each data source is

different, that is, the same building has two different keys in the two different data sources. For instance, the same building ID b001 at source A and ID 233 at source B. A solver (Solver X in Figure 12) can resolve this problem by using a lookup table that has in each row the ID at source A and its corresponding ID at source B.

Once the data have been successfully integrated, the combined data are written into an XML file, as a standard file format for exchanging data. This allows the extracted data to be used by heterogeneous applications. Figure 13 shows an example of the extracted data in XML file format.

```
<?xml version="1.0"?>
<buildings>
  <building>
    <buildingID> b001 </buildingID>
    <address> 24 Ajyad Street </address>
    <height> 34 </height>
    <numOccupants> 340 </numOccupants>
    <yearBuilt>1990</yearBuilt>
    <use> commercial </use>
  </building>
  <building>
    <buildingID>b002</buildingID>
    <address> 567 Umm Al-Qura Street </address>
    <height> 20 </height>
    <numOccupants> 210 </numOccupants>
    <yearBuilt> 1975 </yearBuilt>
    <use> commercial</use>
  </building>
  <building>
    <buildingID> b003 </buildingID>
    <address> 92 Ajyad Street </address>
    <height> 9 </height>
    <numOccupants> 36 </numOccupants>
    <yearBuilt> 2001 </yearBuilt>
    <use> residential </use>
  </building>
</buildings>
```

FIGURE 13. Example XML data retrieved from the data warehouse

A user can either view the extracted data using a web browser or download the data as XML files. The extracted data can then be imported into an urban design application. Another option is to have the requested data automatically shipped to an application using its API without user interference. With this option, the application sends data requests directly to the server and imports the retrieved data directly.

5.3.4 Choosing an Efficient Software Platform

The fourth task is to determine an *efficient software platform* for implementing the data model and the data warehouse for any city or municipality. Several criteria should be used to determine if candidate platforms adequately perform the needed operations:

- *ease of use*, which should be investigated at both ends, by both data providers and urban designers,
- *platform-independent* (like Windows, Macintosh, or Unix), because various data providers and users may use different system-platforms,
- support for object-oriented technology, and
- *web-enabled*, since the data warehouse is web-based.

The software platform to use for this research is investigated in more detail in Chapter 9.

5.3.5 Prototype Implementation and Validation

The fifth major task of this research is implementing both the data model and the data warehouse as a first prototype. This task is explained in detail in Chapter 9.

The prototype can be used to validate and test the approach. This will be done through simulating heterogeneous data sources and possible heterogeneous applications. Because it is not possible to connect the prototype directly to the data sources in Makkah or modify the applications used by urban designers in the city, data sources are simulated using available data at different local authorities and applications are simulated using commercially available software. This task is also described in detail in Chapter 9.

CHAPTER 6

Urban Design Applications and Their Data Needs

This chapter explores the various possible applications for use in urban design and the data needed to support these applications.

6.1 Urban Design Applications

Common urban design applications that need to be supported by the intended data model are discussed briefly in this section.

6.1.1 Visualization

A major concern in urban design has been how to visualize the form of an existing or future urban environment. 3D modeling offers urban designers a useful tool for viewing the visual configurations of an existing urban pattern or understanding the impact of a proposed urban design. 3D urban models can be used to generate 3D and 2D views of an urban area. Furthermore, photo-realistic rendering technology simulates existing physical urban characteristics. Controlling the viewing position in an urban model helps the designer to capture desired scenes.

Recently, Virtual Reality technology has gone one step further than 3D modeling by providing real-time interactive simulations of the visual features of an urban area. This technology enables urban designers to navigate around or through an urban space interactively.

The levels of resolution needed for different forms of visualization vary from overall massing for larger areas down to detailed facade studies. For the overall massing of buildings, only building footprints and heights are needed. For detailed facade studies, the facades of buildings need to be photographed, then texture-mapped onto an urban 3D model.

6.1.2 Analysis

Analyzing an existing urban area or a proposed urban development is another important application. Integrating the geometric and non-geometric information of various entities in an urban setting facilitates both spatial and statistical analysis. For example, land-use information of a building can be integrated with the building's geometry. This integration provides a rich spatial database for discovering and understanding land-use patterns within urban environments.

The 3D model of the city of Glasgow created at ABACUS (Strathclyde University), described in Section 2.1.2.2, is an example of this application. This model indexes building information and land-use information, which can be used to perform quantitative and spatial analysis and to visualize changes to individual buildings and urban spaces [Day 1994].

Housing analysis is another important application. Urban designers and developers need to understand and analyze the existing housing stock and conditions. This includes studying housing units according to their location, type, condition, and capacity. Such analysis is very important for providing solutions to current and future urban housing problems.

Another application may analyze the locations and catchment areas of current urban facilities or services like schools, hospitals, and shops. Suppose an urban developer wants to build a new school for a given urban area. First, catchment areas of available schools should be calculated and visually represented. The developer can then use this

analysis to see where schools are needed. This analysis also facilitates choosing an optimum location for the new school.

6.1.3 Simulation

Simulation is another important functionality that should be supported by the data model. Various kinds of simulations for urban design applications have been implemented. A well-known example is simulating the vehicular movement in an urban area. Mathematical models and algorithms are used to develop a simulation based on input such as geometry of roads, direction of movement, traffic signal locations, origin/destination of vehicles, and land-use. Such a system is an essential tool in urban transportation planning.

Another example application is the simulation of pedestrian circulation in urban areas. Some of the data needed to support this application include the detailed description of human movement behavior, pedestrian characteristics, origin/destination of pedestrians, geometry of walkways and paths, and location of facilities and services. All of these simulations help urban designers understand an existing system or test a proposed one.

6.1.4 Decision Support

Decision Support Systems (DSS) are becoming important tools in urban design, given the complexity of today's urban environments. Basically, a Decision Support System is based on understanding the expected results of a possible decision and answers "what if" questions. A DSS can be developed to check the results of a certain addition or change in an urban environment. Suppose an urban developer wants to see the effects of putting a building with certain characteristics (like a particular number of floors) in a specific location. A DSS can be used to check the effects of placing such a building in that location, and the likely influence of the building on its surrounding streets. For example, the system may show that the building could cause traffic jams on one of these streets. Additional feedback from the system may be that the building should not exceed 10 floors given the regulations and topography of the land. In short, a DSS can be used to test the effect of certain important decisions on urban environments.

6.1.5 Collaboration

Networked settings using the Internet or Intranet offer excellent environments for collaboration among different participants in design and planning. This is an especially important issue in urban design, which often involves the participation of people from multiple disciplines. For example, urban 3D models can now be easily visualized and criticized on-line by multiple "stakeholders". Urban 3D models are

becoming increasingly popular for urban planning and design since they provide an environment to support public participation in judging future developments of urban environments [Day 1994].

6.2 Data Needed

This section describes the data needed to support the urban design applications identified in the previous section.

Urban designers need data in the following categories: topography, water systems, green areas (vegetation), land parcels, built-up volumes, movement network, and administrative and political boundaries [Dave & Schmitt 1994]. A comprehensive city database should include information about zoning, administrative and political boundaries, land parceling, geometry of buildings and their attributes, accessibility, monuments and historic buildings, landmarks, open spaces, traffic network, facilities and amenities, topography, water system, landscaping, and utilities, including water, gas, electricity, and telecommunication [Dave & Schmitt 1994] & [Yin & Williams 1995] and [Jiang 1997].

The following is a detailed explanation of the data needs to support urban design applications.

6.2.1 Zones and Other Bounded Areas

Cities are divided into administrative subdivisions such as neighborhoods, which generally consist of a group of buildings, streets, and open spaces. Data that represent a neighborhood should minimally include the neighborhood name and its boundaries. Other kinds of political or administrative boundaries are municipal and census tracts, and urban planning zones. These bounded areas are important in performing various types of analysis in urban development and planning.

6.2.2 Blocks

A block is a group of land parcels and/or buildings surrounded by streets. Block data should include.

- block ID number,
- parcels within block boundaries, and
- surrounding streets.

6.2.3 Land Parcels (Lots)

To define ownership of land, blocks are divided into subdivisions called parcels or lots. A block consists of one or more land parcels. Each land

parcel defines the boundaries of land ownership. A parcel can be owned by a government, an individual, or a business.

Data needed to represent a land parcel should include.

- parcel ID number,
- address,
- boundaries,
- block number,
- neighborhood or/and district,
- owner ID number
- owner name (first, middle, last)
- owner address
- user(s) information,
- buildings located within the parcel (if built),
- slope,
- value,
- soil characteristics,
- elevation,
- use of land, and
- surrounding parcels/buildings/streets.

6.2.4 Buildings

Buildings are the most important objects in an urban environment because they constitute the physical form of an urban environment. In a

developed urban area, a land parcel may include a single building or a group of buildings.

There are two main kinds of building data that are needed to support urban design applications (see Figure 14):

- *Geometric data* that include all information describing a building's geometry and its visual features. These data also include the geographic (spatial) location of the building. These data support visualization and spatial analysis applications, and include building outline, height, and facades. More complex buildings may have to be described as a collection of volumes.
- *Non-geometric data* that include building identification (ID) number, address, use, name, type, condition, ownership, materials, number of floors, number of occupants, character, value, accessibility, architect, year built, and if residential number of dwellings and number of rooms per dwelling. These data offer a source for important applications such as land use analysis, evaluating current building conditions, and generating different queries about buildings. Street address data have to include building number, street name, street type, direction, unit (apartment or suite) number, city, and zip code. The street address data provide important input for performing address-matching algorithms. Address matching is a technique used in Geographic Information Systems for finding or analyzing buildings based on their street address. Another important use of

these data is to calculate the algorithm for the shortest path from one address to another.

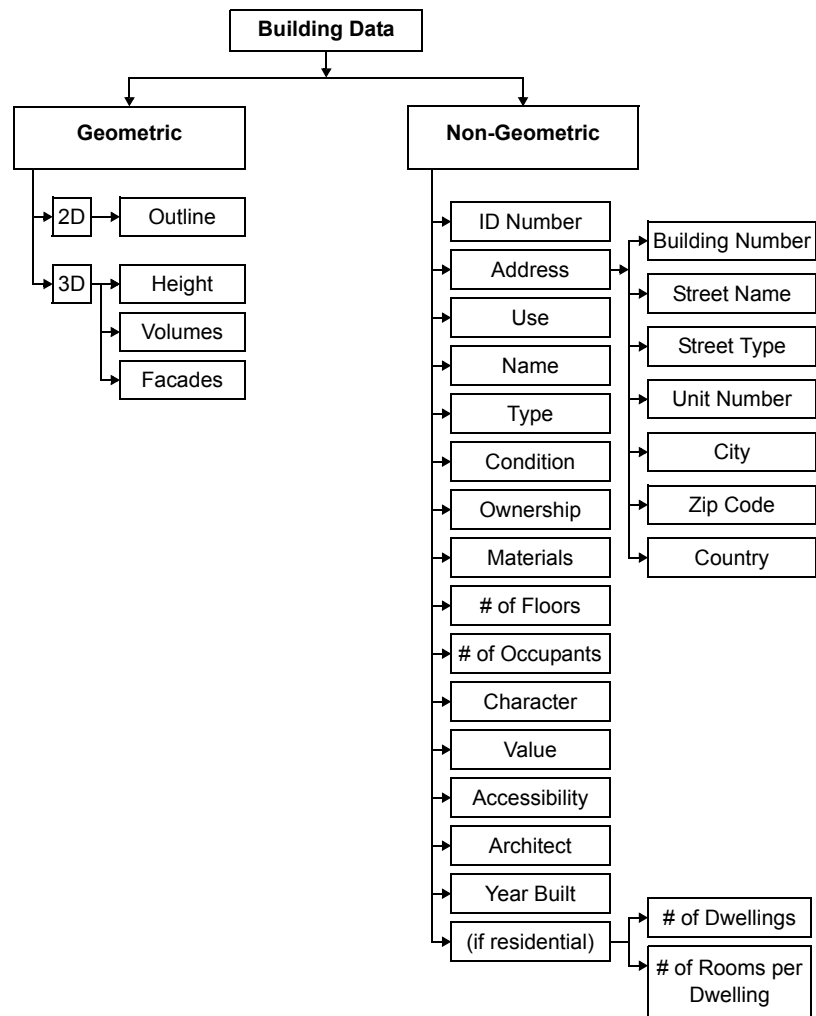


FIGURE 14. Building data

6.2.5 Movement Network (Vehicular and Pedestrian)

The second most important component of an urban area is the traffic (movement) network, which facilitates vehicular and pedestrian movements from one place to another. A network is a system of connected linear features (like streets, paths, and walkways) through which resources (moving objects) flow.

A movement network model contains a number of elements (Table 2):

- links,
- turns,
- stops,
- centers, and
- barriers.

Once a movement network is modeled, it can be used as input to various algorithms. Some examples of these algorithms are the following:

- Route or path finding (to find out the optimal paths for moving resources),
- Sequencing (to investigate the optimal order for pick-up/drop-off sequences),
- Clustering or allocating (to view how resources are distributed among centers), and
- Time window routing (e.g. school busses) [Gant & Tita 1997].

Network Element	Definition/Function	Examples	Attributes
Links	conduits for movements	streets, paths, or walkways	<ul style="list-style-type: none"> - Two-way impedances: time to traverse a segment; represents resistance of movement - Resource demand: number or amount of resources on every arc and stop in network
Turns	set of possible turns	street intersections	<ul style="list-style-type: none"> - Impedance, such as turning time - Restrictions, such as no left turn
Barriers	prevent movement between links and resources not allowed to flow through a barrier	closed street	<ul style="list-style-type: none"> - Location of barrier - Kind of barrier
Centers	locations that receive or distribute resources	schools, or fire stations	<ul style="list-style-type: none"> - Resource capacity, such as parking spaces - Impedance limit, such as maximum distance or time between a center and a link
Stops	locations on a route to pick-up or drop-off resources	bus stops, or warehouse	<ul style="list-style-type: none"> - Demand for resources to be transported, such as students, or products

TABLE 2. Movement network elements

6.2.6 Open Spaces

An open space is an unbuilt area defined by its surrounding objects such as buildings and/or streets. The data needed to represent an open space include

- space name,
- space layout,
- space functionality (use),
- objects surrounding the space (buildings or streets), and

- objects inside the space (vegetation and other landscape elements).

6.2.7 Vegetation and Landscape Features

Vegetation and landscape elements are used to vitalize urban spaces. These elements include

- trees (including individual trees, trees in rows, and woodland),
- shrubs,
- grass,
- rough grasslands and reeds,
- boulders,
- rocks and scattered rocks,
- scrubs,
- orchards,
- gravel,
- mud, and
- sand.

Data that represent a landscape or vegetation feature include the following:

- landscape/vegetation type (tree, shrub, or grass),
- location,
- physical dimensions,

- kind of subcategory (like the type of tree), and
- height (for trees).

6.2.8 Topography (Terrain)

Land forms of cities vary from flat to mountainous. In cases where cities are located in mountainous areas, topography becomes is an important issue. Topography (terrain) has been represented with two techniques:

- contour lines (each line follows an equal elevation value), and
- elevation points (each point represents a height) in an array covering an area of interest.

These representations have been supported digitally in the following formats:

- Polygons (representing contour lines)
- Points (representing elevation points)
- Digital Terrain Models (DTMs) that represent a land surface. Two types of DTM data models are commonly used in GIS systems: a grid-based data model, and a triangulated irregular network (TIN) model.

6.2.9 Hydrography (Water System)

In cities where there are rivers, canals, falls, lakes, and other water features, these objects have to be represented. This representation should include

- type of water object,
- location,
- direction of water flow,
- physical dimensions (geometry), and
- characteristics of water.

6.2.10 Facilities/Services

Facilities and services are essential resources of an urban environment and include

- public transportation,
- food and beverage (grocery) stores,
- cafeterias and restaurants,
- public drinking water,
- educational (primary, middle, and secondary schools, colleges, and universities),
- public telephones,
- guiding services (information centers and sign posting),
- postal service,
- shopping facilities,

- health services (clinics and hospitals),
- security and safety (police station, fire station, and ambulance centers),
- public parking lots,
- accommodation facilities (e.g. hotels and motels),
- public parks and recreational places,
- religious facilities,
- money exchange facilities,
- laundry washing facilities,
- and public toilets.

The data that represent each facility/service should include

- location (address) of the facility/service,
- type (one of the listed above),
- name,
- catchment distance,
- geometry (outline), and
- capacity.

6.2.11 Utilities

Utilities are essential components of the infrastructure of cities. These include

- telecommunication (e.g. telephone lines and television cables),

- electricity,
- water supply,
- sewage,
- gas,
- trash removal and disposal, and
- storm water drainage.

Digital representation of these utilities should include

- location of a utility component,
- type of utility (electrical, gas, or telecommunication),
- name (e.g. manholes or fire hydrants)
- physical dimensions (geometry),
- material, and
- utility provider.

6.2.12 Moving objects

Urban environments include moving objects like people and vehicles. Vehicular movement and pedestrian circulation are very important aspects of urban design and planning. In transportation planning, traffic movement (vehicular and pedestrian) is analogous to the “software” and the traffic network (streets, paths, or walkways) is analogous to the “hardware” in computer design.

Data that represent a vehicle should include

- type of object (car, bus, or train),
- make and model,
- number of passengers,
- physical dimensions, and
- pattern of movement, including origin/destination.

Data that represent a person should include

- age,
- nationality,
- marital status,
- disability (if any),
- level of education,
- income,
- car ownership, and
- pattern of movement, including origin/destination.

Vehicular and pedestrian movements are affected by other elements and their characteristics such as street type, speed limit, and degree of congestion. These factors should be added to networks to realistically model movement through the traffic network described in Section 6.2.5. For instance, socio-economic factors such as age, income, and gender are very important inputs in human behavioral modeling and simulation applications.

The following tables (Table 3 & Table 4) cross-reference various urban data with the types of urban design applications that need them. The tables show that the same data may be needed by different

applications. For instance, land-use data are needed for traffic simulation as well as land-use analysis applications. The tables also show that an application may require data in multiple categories and from multiple sources. For instance, the housing demand application requires the number of occupants in each building, the number of housing units within the building, and the building condition.

I	APPLICATIONS (Part 1)																		
	3D visualization	address matching	vehicular traffic simulation	facade studies	shortest path analysis	pedestrian movement simulation	facilities distribution analysis	housing stock inventory	VR visualization	circulation analysis	land-use planning/management	urban redevelopment analysis	building views analysis	utility needs analysis	air pollution analysis	sound pollution analysis	human behavior simulation	emergency response analysis	vegetation analysis
DATA																			
3D building models	X			X					X			X	X						X
origin/destination of cars			X							X					X	X			X
street address		X			X					X								X	
facades texture maps	X			X					X			X							
origin/destination of pedestrians						X				X						X			
housing units capacity							X	X						X					
street segments			X							X									
land parcel boundary	X	X						X		X	X				X				
traffic network maps	X	X	X	X	X	X			X	X	X	X			X			X	X
building conditions												X							
building footprints	X	X		X				X	X			X	X					X	X
utility maps														X					
building resident characteristics						X	X												
vegetation	X																	X	
sun movement				X															X
land owner information																			X
building age (year built)																		X	
right of way			X			X													
attractions						X				X								X	
zoning																			
curb lines	X																		
sidewalks						X			X										
street intersections			X							X									
traffic control facilities			X							X									
facilities and services							X			X									X
topography	X		X	X					X	X								X	X

TABLE 3. Data needs for different urban design applications (Part I)

II	APPLICATIONS (Part 2)																								
DATA	impact assessment	vehicular & pedestrian routing	emergency services planning	major projects review	facility siting and design	natural resource management	building permits	business licensing	new business impact	site analysis/planning	vacant lot inventories	economic development analysis	resource allocation	urban management & monitoring	flood plain analysis	environmental resource analysis	soil analysis	resource planning decision	addressing	usage patterns	land-use analysis	real estate research	housing conditions/inspections	physical planning analysis	
3D building models	X									X				X										X	
origin/destination of cars		X																							
street address		X							X										X						
facades texture maps																									
origin/destination of pedestrians		X			X																				
housing units capacity					X				X					X											
street segments		X																							
land parcel outlines		X					X		X	X	X		X	X		X	X		X			X			
traffic network maps				X					X	X	X		X	X	X	X					X			X	
building conditions										X													X		
building footprints				X	X		X		X	X	X					X			X			X		X	
utility maps										X															
building occupants info											X												X		
vegetation						X				X						X									
sun movement																X									
land owner information											X											X			
building age (year built)										X															
right of way																									
attractions										X															
zoning				X						X								X			X				
curb lines																									
sidewalks				X																					
street intersections				X																					
traffic control facilities					X																				
facilities and services					X		X			X															
topography						X				X						X									
land-use				X						X											X	X	X		

TABLE 4. Data needs for different urban design applications (Part 2)

CHAPTER 7

Unified Object-Oriented Data Model

A major task in this research is developing a unified object-oriented data model that captures all data listed in Section 6.2. This model is required for the data warehouse and enables it to integrate the data coming from heterogeneous sources.

7.1 Object Models or Schemas

As discussed in Section 5.3.3, the urban data warehouse must be able to extract data from heterogeneous sources, to integrate the data using a unified data model, and to provide input to heterogeneous urban design applications or answer user queries through a uniform interface. The unified data model (schema) integrates specifically geometric and

non-geometric data that are currently typically distributed over heterogeneous, non-coordinated sources (like a DBMS and CAD files). Furthermore, an application that has an interface able to receive data from a warehouse supporting a specific data model will be able to receive data from any warehouse supporting that model, that is, will be able to collect data for any city or region maintaining such a warehouse.

There are good reasons to demand that this unified data model be object-oriented. Research and practical applications over the last decades have amply demonstrated that object-oriented representations are uniquely able to capture the attributes of designed artifacts in a natural and computationally efficient fashion. Furthermore, object-oriented programming has by now established itself as the paradigm of choice for the development of robust, extensible, and reusable software [Meyer 1988] (see also Section 4.2). An object-oriented representation is therefore not only desirable in its own right, but also facilitates data import to object-oriented applications as they are coming into wide use.

An object-oriented (OO) representation is a collection of objects with attributes, where some attributes may be relations with other objects. The attributes that an object can have are typically determined by the class to which the object belongs. Classes can inherit attributes from super classes and pass these attributes to all objects belonging to the class (these objects are often called instances of the class). An *object model* or *schema* is the collection of all classes describing the universe of discourse for an application or group of cooperating applications. All data handled by an application are captured by objects that are

instances of classes in the schema. The unified object model underlying our data warehouse is a schema in this sense: its classes define the types of objects that can be instantiated to import data (via object attributes) to urban design clients.

Over the last decade, various notations have been proposed for the documentation and communication of object models. Among these, the notation developed by Rumbaugh [Rumbaugh et al. 1991] has become particularly popular and is now an integral part of the Unified Modeling Language UML [Booch et al. 1999]. UML has become the standard language used for object-oriented software development [Naiburg & Maksimchuk 2001]. This chapter uses Rumbaugh's notation, via UML, to develop and document the object model underlying the data warehouse; examples are shown in the figures below.

The schema presented in this chapter is a collection of classes each of which collects all (geometric and non-geometric) attributes of an urban object. The schema has been developed by first looking at the data available in heterogeneous sources (Chapter 3). A DBMS, for instance, provides data in the form of flat tables with multiple columns recording various non-geometric attributes of urban objects. A CAD file may represent urban areas or building outlines as 2-dimensional polygons defined by the coordinates of their corners. A GIS source typically consists of flat tables linked with 2D geometries to describe buildings for instance. In short, the data models used by the data sources are structured collections of value attributes. The class attributes in this

schema either capture these attributes directly or through associated classes.

7.2 Schema Requirements

A schema that integrates attributes of this kind so that heterogeneous applications can import them must satisfy the following requirements.

First, the schema must be comprehensive with respect to data needed to support urban design applications. The sources may not include all the needed data at the present time, but the model has to be comprehensive enough to capture these data so that they can be provided, in the future, by the same or new sources. The schema also has to be complete with respect to the data available at the heterogeneous sources. That is, it has to capture all data available at the sources even if the applications currently used in Makkah do not need them.

Second, the classes of the schema must define objects that are meaningful for the domain of urban design. They should represent objects such as buildings, land parcels, and topography as introduced in Chapter 6.

Third, the schema has to be simple enough to facilitate parsing and interpretation by various urban design applications. To achieve simplicity, the following criteria have to be satisfied.

- The schema captures no behavior; that is, classes have no method attributes, and objects that instantiate them are strictly passive data repositories.
- As a data warehousing requirement, each object has to have a unique global identifier to allow data integration from heterogeneous sources. For this reason, each class must have a key attribute (ID), which distinguishes each urban object globally across the various sources.
- Many-to-many relations between objects are kept to a minimum. When data are mapped from the unified model to the internal model of an application, many-to-many relations pose a problem for the mapping routine because it may encounter the same objects several times. In order to avoid duplicate mappings, the routine has to keep track of the objects already mapped. Many-to-many relations cannot be completely avoided (see the building owner example in Section 7.3.5), but the model can at least try to keep them to a minimum.
- The schema represents simplified object geometries. This representation has to be sufficient to capture geometries currently available in data sources and able to describe volumetric massing models. It does not need to capture detailed design features of objects.

- Subclass (inheritance) relations are used sparingly; they are needed only to provide some flexibility in describing the various geometries that may be associated with an object like a building.
- Multiple inheritance is strictly avoided to simplify parsing and interpretation, especially in applications written in languages that do not support multiple inheritance (such as Java).

7.3 Urban Object Models

The following classes have been developed to describe various objects in urban environments. The UML notation used in the following diagrams is explained in the Appendix.

Note that the diagrams below are class diagrams that may show one-to-many or many-to-many relations between classes. Most often, this means that an object instantiating a class has multiple relations to several objects each of which instantiates the same class, but is a distinct instance of that class. For example, a Polyline2D has several associated Point2D objects, each of which is an instance of the Point2D class, but differs from the other points by its coordinate values.

7.3.1 Geometry Classes

Object geometries often rely on the same geometric elements across classes. The model therefore separates the description of geometric attributes from non-geometric attributes and uses separate geometry classes to represent geometric elements. Objects with shared geometries can then be associated with instances of these classes.

The various geometries of urban objects fall into two basic classes, representing respectively, 2- and 3- dimensional elements: Shape2D and Solid. The Shape2D class (Figure 15) is designed to capture two-dimensional (2D) shapes of urban objects. The Shape2D has an attribute that captures the elevation of an urban object. This attribute can be used to place an object at some precise location. The Shape2D is divided into two subclasses: an OpenShape2D and a ClosedShape2D. The OpenShape2D is divided into two subclasses: a Polyline2D and an Arc2D. The Polyline2D is associated with many Point2D objects, representing its corners. The ClosedShape2D class is divided into three subclasses: a Polygon, an Ellipse, and a Circle. The Polygon2D is associated with many Point2Ds, representing its corners. The Point2D has two attributes which capture a pair of double-precision coordinates in the order X and Y. The Ellipse attributes capture the length of axis 1, the length of axis 2, the angle, and the center point of an ellipse. The Circle attributes capture the radius and the center point of a circle.

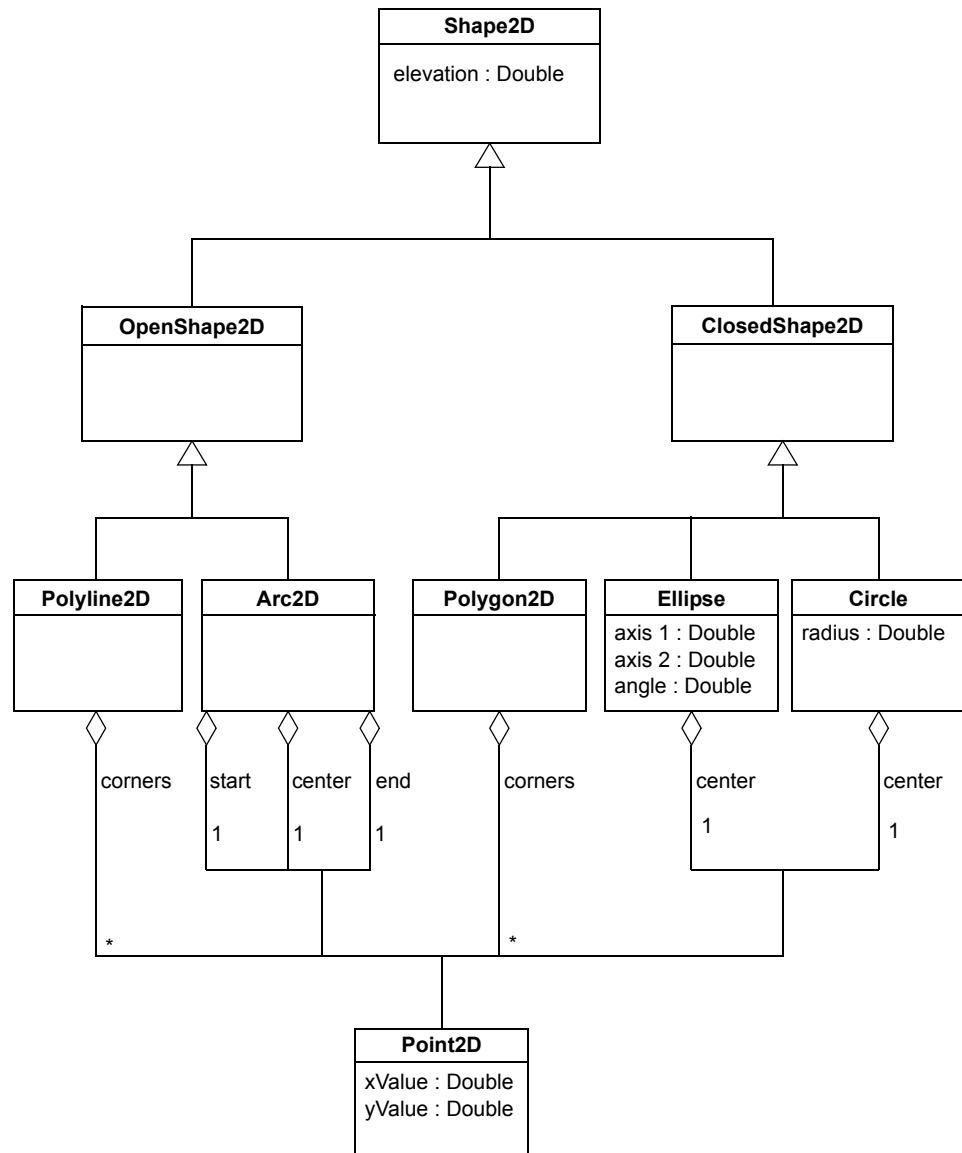


FIGURE 15. Shape2D Class

The Solid class (Figure 16) is designed to capture three-dimensional (3D) shapes (forms) of urban objects. The Solid has an attribute that

captures the elevation of an urban object. The Solid is subdivided into two subclasses: a Prism and an ExtrudedSolid. A Polygon3D class describes 3D polygons so that a prism can be described by several associated 3D polygons representing its faces. Each Polygon3D is associated with many Point3Ds, representing its corners. Each Point3D has three attributes which capture its double-precision coordinates in the order X, Y, and Z. The ExtrudedSolid is associated with one ClosedShape2D described above and has a height attribute. From this information, its faces can easily be constructed for display purposes.

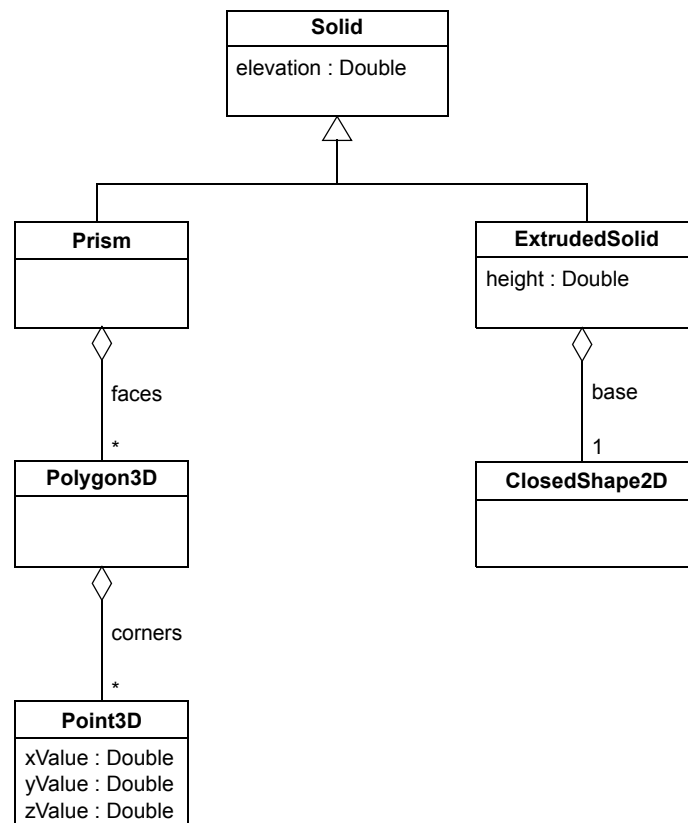


FIGURE 16. Solid Class

7.3.2 Zone

The *Zone* class captures all attributes of an administrative or political zone in an urban environment listed in Section 6.2.1. Figure 17 shows the *Zone* class. It is associated with a *ClosedShape2D*, described in Section 7.3.1, that captures the geometry of its boundary.

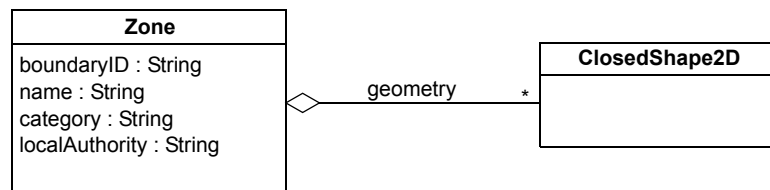


FIGURE 17. Zone Class

7.3.3 Block

The *Block* class captures all required attributes of a block in an urban environment listed in Section 6.2.2. Figure 18 shows the *Block* class. It is associated with a *ClosedShape2D*, described in Section 7.3.1, that captures the geometry of its boundary.

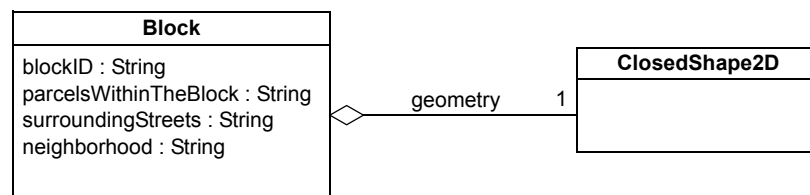


FIGURE 18. Block Class

7.3.4 LandParcel

A *LandParcel* class captures all non-geometric and geometric data listed in Section 6.2.3 for objects of this kind. Figure 19 shows this class. It is associated with a *ClosedShape2D* that captures the geometry of its boundary. It is also associated with an *Owner* and *Deed* class.

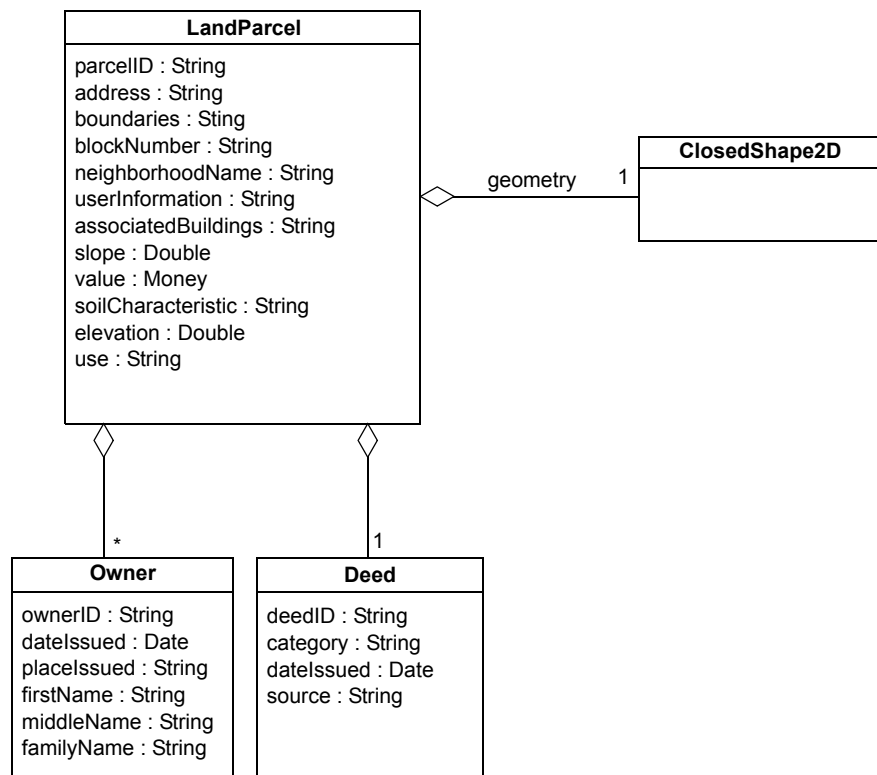


FIGURE 19. LandParcel Class

7.3.5 Building

The *Building* class captures all attributes of a building in an urban environment as listed in Section 6.2.4. Figure 20 shows the Building class. It is associated with a BuildingGeometry class that captures the geometry of a building as a collection of Solids described in Section 7.3.1.

The BuildingGeometry can capture the volume or mass of buildings at a relatively abstract level. For instance, it can represent the Holy Mosque located in the central area of Makkah as shown in Figure 21. The Mosque has the most complex form of buildings in central Makkah. ExtrudedSolids describe most parts of the Mosque, the building surrounding the courtyard as well as the Kabah located in the center of the courtyard. Prisms are used to represent the top of the minarets.

As this example shows, the BuildingGeometry model is not meant to provide a very detailed and sophisticated 3D model of a building. Rather, the model is developed to capture geometries of buildings in a form sufficient to support the urban design applications discussed in Section 6.1. Developing a general model for the representation of buildings and other urban objects that would be able to cover the needs of a broader range of urban design applications is beyond the scope of this dissertation.

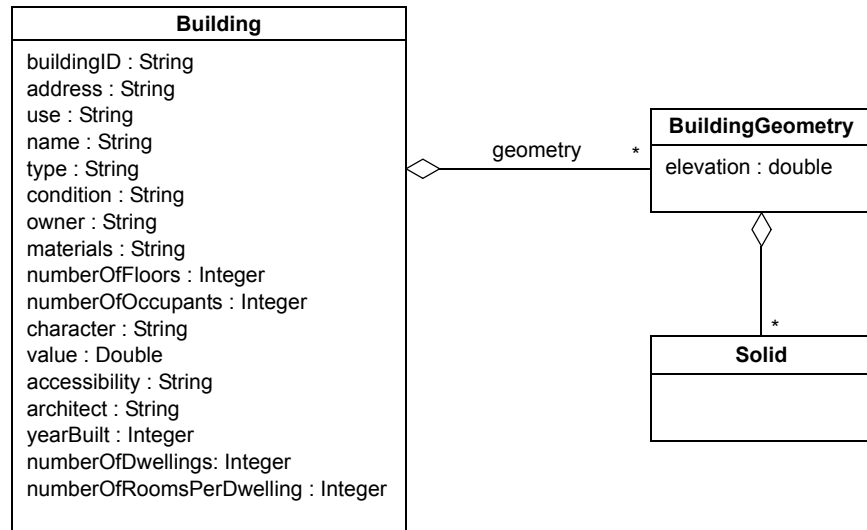


FIGURE 20. Building Class

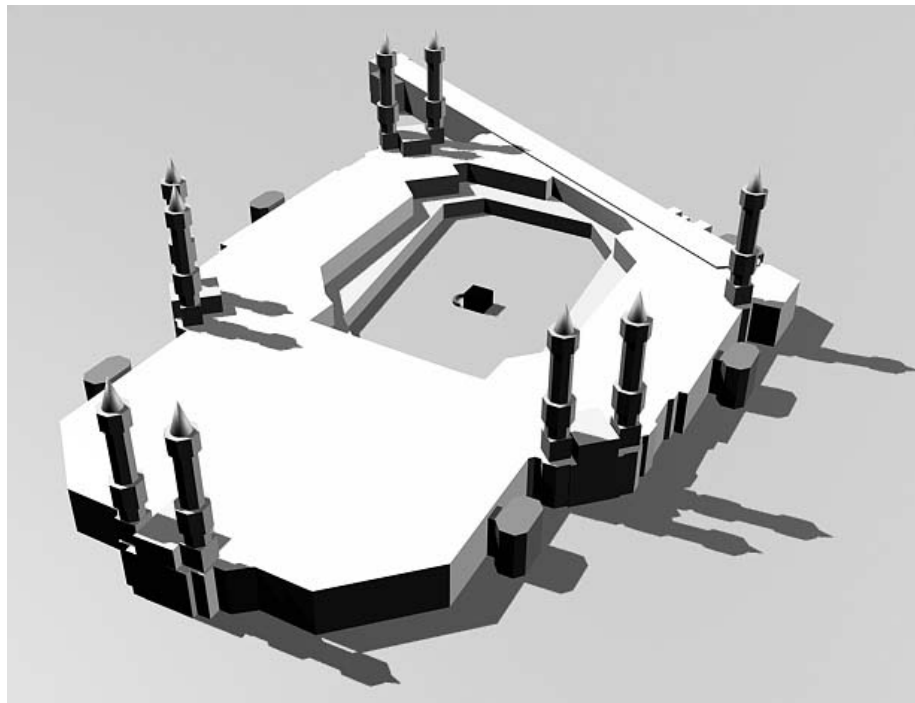


FIGURE 21. Geometry of the Holy Mosque

7.3.6 NetworkElement

The *NetworkElement* class captures all attributes of a component in a vehicular or pedestrian movement network in an urban environment listed in Section 6.2.5. Figure 22 shows the *NetworkElement* class. It is divided into the five subclasses introduced in Section 6.2.5: a *Link*, a *Turn*, a *Barrier*, a *Center*, and a *Stop*. Each of these classes represents a type of object in a movement network. Each class is associated with a *NetworkElementGeometry* that captures the geometry of that class.

The geometry of a network element can fall into one of two categories: junctions or edges. Therefore, the *NetworkElementGeometry* is divided into two subclasses: a *NetworkEdge* and a *NetworkJunction*. The *NetworkEdge* can be used to represent a linear object like a street and is associated with an *OpenShape2D*, described in Section 7.3.1. The *NetworkJunction* can be used to represent a node such as a street intersection and is associated with a *Point2D*.

Note how this pattern circumvents a problem introduced by single inheritance. We cannot really make an *OpenShape2D* a subclass of *NetworkElementGeometry* because it is already a subclass of *Shape2D*. *Point2D* should not subclass *NetworkElementGeometry* either because it may be used to represent also other types of point locations. By introducing *NetworkEdge* and *NetworkJunction* subclasses of *NetworkGeometry*, and associating them with the proper geometry classes, the model arrives at a configuration that serves its

purposes. Similar patterns are used below to capture object geometries that need various combinations of geometry classes.

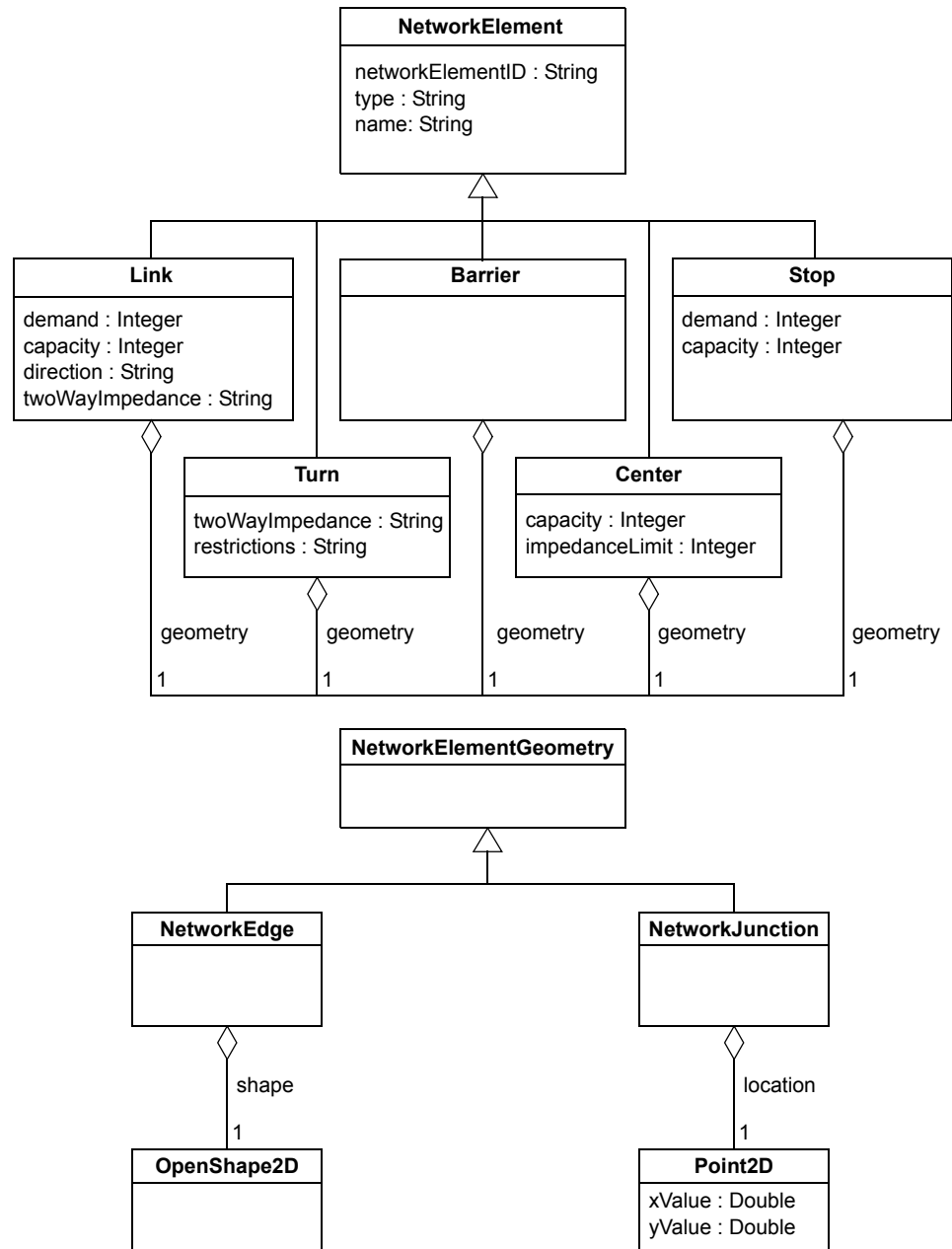


FIGURE 22. NetworkElement Class

7.3.7 OpenSpace

The *OpenSpace* class captures all attributes of an open space in an urban environment listed in Section 6.2.6. Figure 23 shows the *OpenSpace* class. It is associated with a *ClosedShape2D*, described in Section 7.3.1, that describes the boundary of an open space.

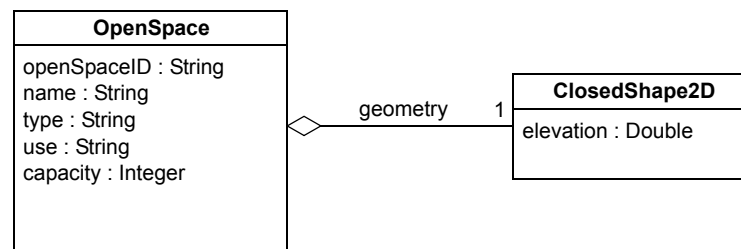


FIGURE 23. *OpenSpace* Class

7.3.8 Landscape

The *Landscape* class captures all attributes of a vegetation or landscape feature in an urban environment (Section 6.2.7). Figure 24 shows the *Landscape* class. It is associated with a

LandscapeGeometry class that captures the geometry of a vegetation feature or a landscape feature.

The LandscapeGeometry is classified into two subclasses: a LandscapeArea and a LandscapePoint. The LandscapeArea can be used to represent an area of a vegetation or landscape feature like a grassy area; it is associated with a ClosedShape2D to capture its boundary. The LandscapePoint can be used to represent a point feature like a tree and is associated with a Point2D that captures its location.

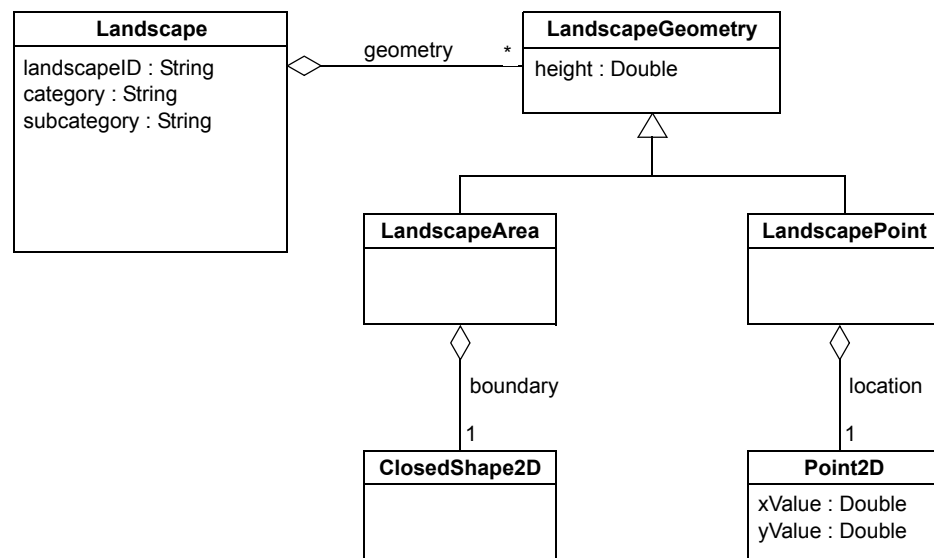


FIGURE 24. Landscape Class

7.3.9 Topography

The *Topography* class captures all attributes of topography (terrain) features in urban environments listed in Section 6.2.8. Figure 25 shows the Topography class. It is associated with a TopographyGeometry class that captures the geometry of a topographic feature in an urban area.

As explained in Section 6.2.8, the geometry of a topographic feature can be described by an array of 3D points or a set of contour lines. For this reason, the TopographyGeometry is divided into two subclasses: a PointArray and a ContourLines. The PointArray represents an array of elevation points in an urban area and is associated with many Point3Ds described in Section 7.3.1. The ContourLines class represents a traditional contour map, where the contours are represented by polylines.

7.3.10 Hydrography

The *Hydrography* class captures all attributes of a hydrography feature in an urban environment (Section 6.2.9). Figure 26 shows the Hydrography class. It is associated with a HydrographyGeometry class that describes the geometry of a hydrography feature in an urban area.

The HydrographyGeometry is divided into three subclasses: an AreaFeature to represent features such as lakes, a LinearFeature to represent features such as rivers, and a PointFeature to represent

features such as water wells. Each one of these subclasses is associated with an appropriate geometry class to capture its geometric attributes.

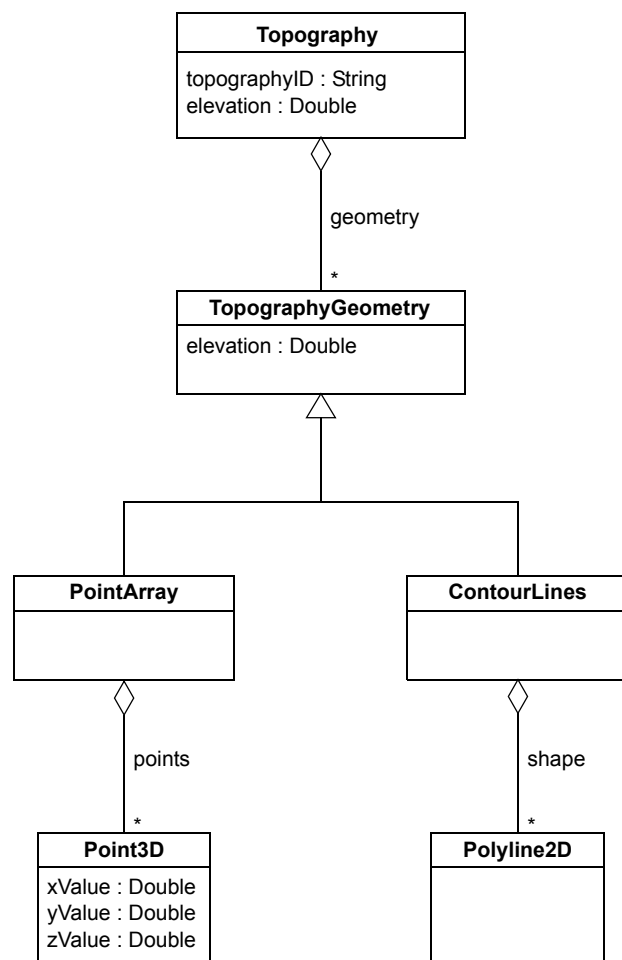


FIGURE 25. Topography Class

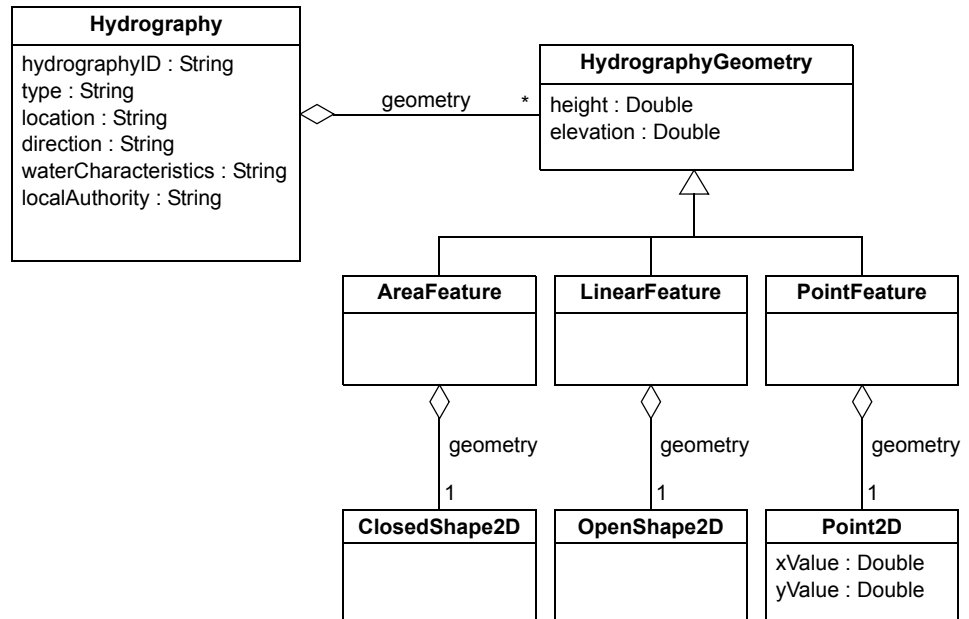


FIGURE 26. Hydrography Class

7.3.11 Facility

The *Facility* class captures all attributes of a facility or service in an urban environment listed in Section 6.2.10. Figure 27 shows the Facility class. It is associated with a FacilityGeometry class that describes the

geometry of a facility feature in an urban area. The FacilityGeometry class is divided into two subclasses: a FacilityArea and a FacilityPoint.

The FacilityArea can be used to represent a facility area like a public parking lot or a public recreational area and is associated with a ClosedShape2D to describe its boundary. The FacilityPoint can be used to represent a facility location (point) like a public telephone or a bus stop and is associated with a Point2D to capture its location.

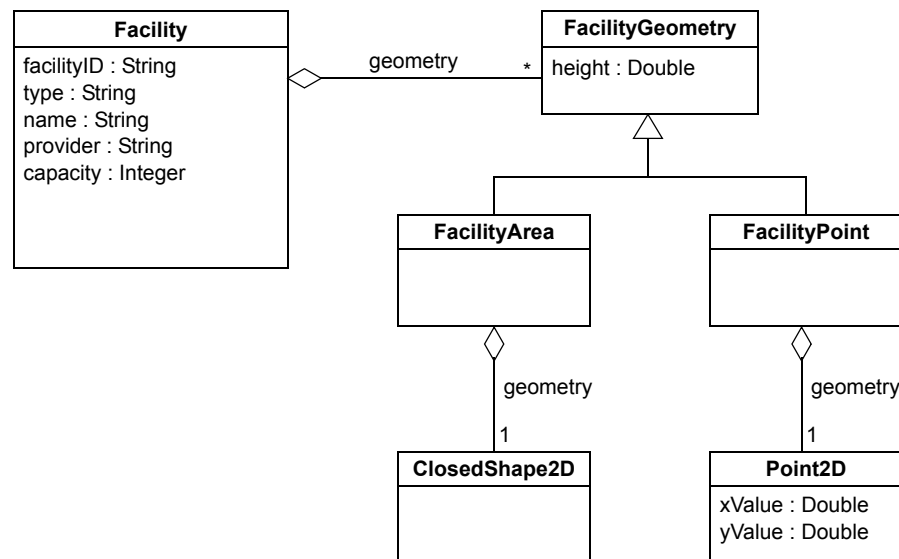


FIGURE 27. Facility Class

7.3.12 Utility

A *Utility* class captures all attributes of a utility feature, which is part of a utility network in an urban environment (Section 6.2.11). Figure 28 shows the *Utility* class. It is associated with a *UtilityGeometry* class that describes the geometry of a utility feature.

The *UtilityGeometry* is divided into two subclasses: a *UtilityEdge* and a *UtilityJunction*. The *UtilityEdge* can be used to represent linear features such as water pipes or electricity cables and is associated with an *OpenShape2D* to capture its geometry. The *UtilityJunction* can be used to represent nodes such as manholes or electricity stations and is associated with *Point2D* to capture its location.

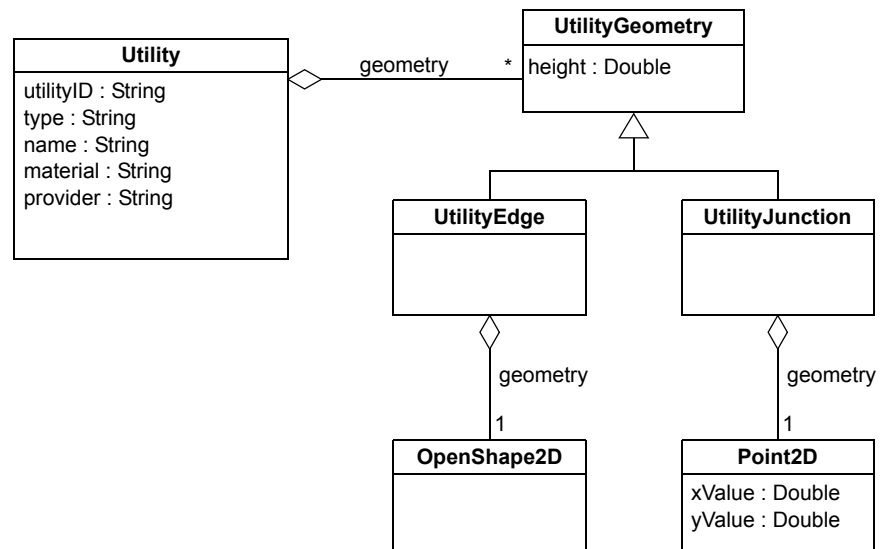


FIGURE 28. Utility Class

7.3.13 MovingObject

The *MovingObject* class captures all attributes of a moving object, including a person or vehicle in an urban environment, listed in Section 6.2.12. Figure 29 shows the *MovingObject* class. It is divided into two subclasses: a *Vehicle* and a *Person*. The *Vehicle* describes a vehicle such as a car, bus, or train. The *Person* class describes a person like a pilgrim visiting the city of Makkah. Both the *Vehicle* and the *Person* are associated with a *MovementPattern* class that describes the origin and destination of movement.

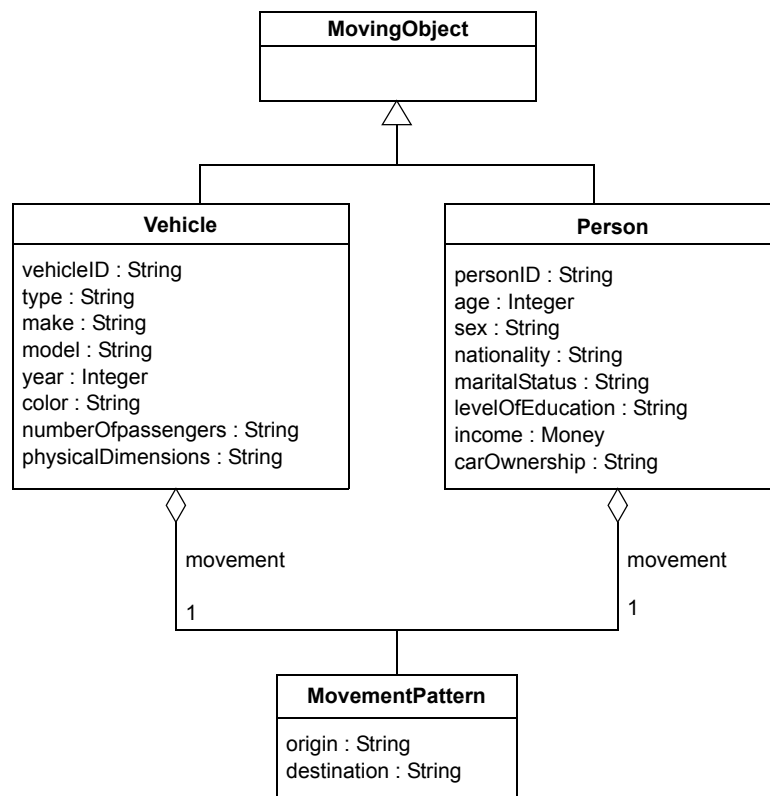


FIGURE 29. *MovingObject* Class

This chapter introduced the unified object-oriented data model that is required for the proposed urban data warehouse. The following chapter describes three major components of the warehouse: Data Access Objects, Extractors, and Solvers.

CHAPTER 8

Data Warehouse Components

This chapter describes three main components of the proposed urban data warehouse: Data Access Objects, Extractors, and Solvers. Each of them performs a specific functionality within the system architecture of the data warehouse. As discussed previously in Section 5.3.3, they work together to deliver the required capabilities of the data warehouse.

8.1 Data Access Objects (DAOs)

A *Data Access Object* (DAO) is a software component that is able to answer queries on objects whose attributes are recorded by heterogeneous sources. The urban data warehouse must contain a DAO for each class in the underlying schema. For instance, a “Building” as an urban object requires a “BuildingDAO” as a DAO. The Building object captures all the attributes of a building in an urban environment.

The BuildingDAO indicates where the values for these attributes are located across heterogeneous data sources.

The DAO provides a query (get) method for each attribute in the associated class. It needs an Extractor for each of the heterogeneous sources to import the data recorded in that source. Accordingly, the DAO selects the appropriate extractor when a specific attribute value is needed and uses it to obtain the value for that attribute.

The DAO represents a design pattern that separates an application from its data sources. This has two advantages. First, the applications are isolated from changes in the data sources; they are, in fact, able to retrieve data from different warehouses, as long as the warehouses support DAOs based on the same schema. Secondly, the data in the sources become available to different applications [Wheeler & Wheeler 2001]. For example, the applications do not have to modify their data import procedures when the source for specific data items changes (although the extractors for this information will have to change). Moreover, applications that are able to import data from the warehouse are also able to import, without further modification, data from any other warehouse that supports the same schema and DAOs.

Take a Building object as an example: It captures among others, the attributes building height, year built, and number of occupants. The building height is stored at data source A, which is a GIS system. The year built and number of occupants are stored in data source B, which is a DBMS. A BuildingDAO knows the location of these attributes and which extractors are needed. In order to obtain the building height, the

BuildingDAO connects to data source A and uses a GIS extractor. To obtain the year built and number of occupants attributes, the BuildingDAO connects to data source B and uses a DBMS extractor (see Figure 30).

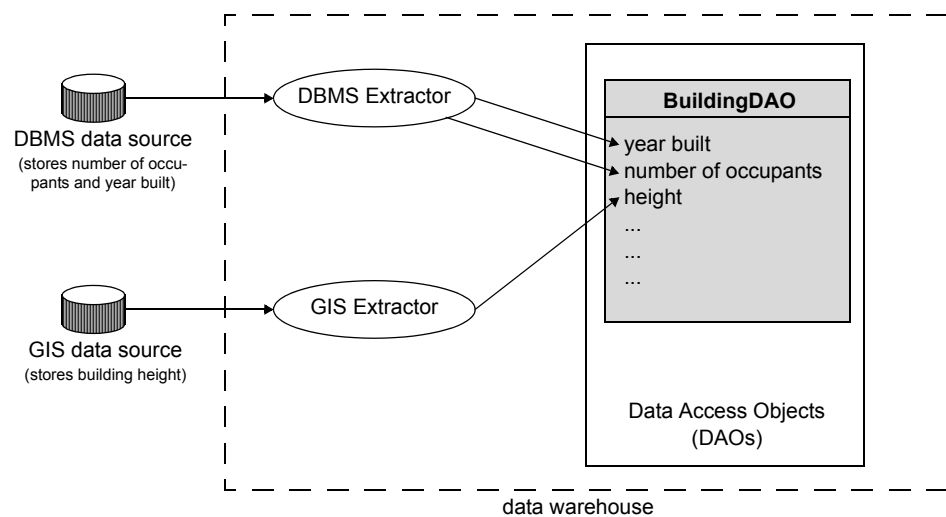


FIGURE 30. Interactions between data sources, Extractors, and DAOs

8.2 Extractors

An *Extractor* is a software component that receives a data request, connects to the appropriate data source, and returns the requested data. As illustrated in Figure 31, an extractor has to be implemented for

each kind of data source, and the data warehouse may have to provide, for example, a DBMS Extractor, a CAD Extractor, and a GIS Extractor.

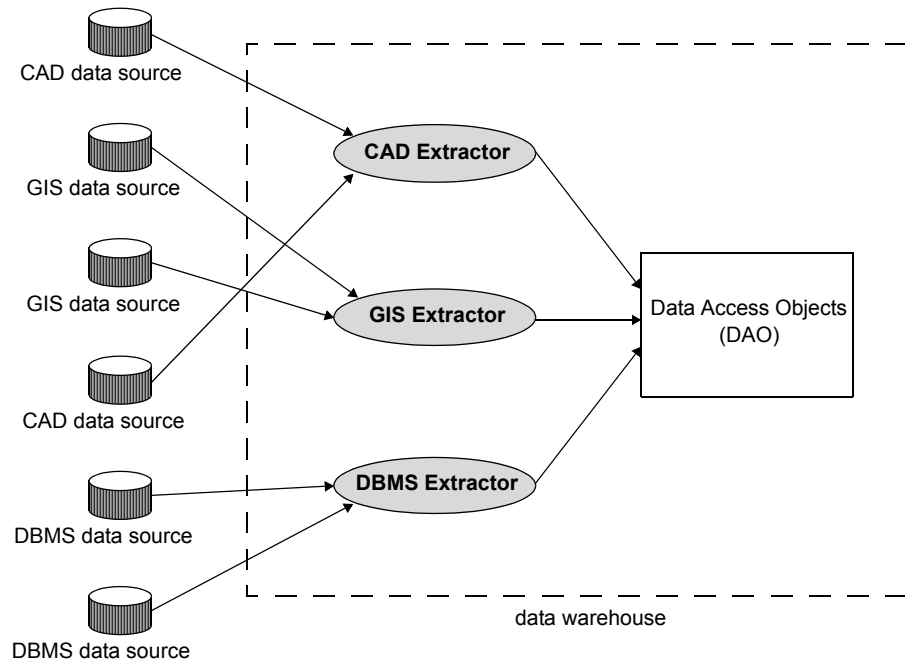


FIGURE 31. An Extractor has to be implemented for each kind of data source

An extractor receives two arguments from a DAO: a data source location and a query. It then connects to the specified data source and returns data based on the specified query. For example, a DBMS extractor takes a URL (Uniform Resource Locator) string and SQL (Structured Query Language) statement. The URL indicates where the (remote or local) database is located. The SQL statement specifies the selection criteria of the requested data. The extractor uses these

arguments to connect to the database and to retrieve the data. It then returns the requested data to the DAO.

The extractors described above extend the wrapper concept to data extraction. A wrapper is a software component that translates data and queries from one model to another [Papakonstantinou et al. 1995a]. In the same sense, an extractor translates data requests from a DAO into source-specific queries, that is, it plays the same role as a wrapper.

8.3 Solvers

A *Solver* is a software component that is able to perform data cleaning, transformation, and integration tasks (see Figure 32). The motivation to introduce solvers into the general data warehouse architecture stems from observations like the following.

Suppose that two different local authorities are using two different GIS map projections that represent the same urban area. A map projection represents locations on the globe onto the flat surface of a map. All map projections distort the shapes of the features being displayed to some degree; this also holds for measurements of area, distance, and direction (Kennedy & Kopp 2001). In this case, the data warehouse needs a process that is able to resolve the data conflict in some

appropriate way. In the data warehouse proposed here, these processes are generally delegated to Solvers.

Another typical data integration problem arises when the key (identifying) attribute for the same object is different at two different data sources; for example, the same building has building ID b041 at a DBMS data source but building ID 3873 at a GIS data source. A solver handling this problem can map the building identification numbers between the two data sources, using a lookup table.

The data warehouse administrator integrates these solvers with the data warehouse architecture as illustrated in Figure 32. The way solvers are selected varies according to the problem the warehouse is handling. In cases such as the map projection problem, the selection of solvers is left to the user; it can be handled by preference features in a user interface through which the user can select the appropriate solver from the available ones whenever such a data integration problem arises. In other cases, like the key attributes problem, the DAO calls and uses the appropriate solver directly whenever this integration problem occurs.

As an example, the Solver that handles the problem of having different key attributes has a method named *Solve* which takes an object ID number as an argument, finds the building ID number at the first data source, and then returns its corresponding building ID number at the second data source. The DAO then uses the right building ID to obtain its data using Extractors.

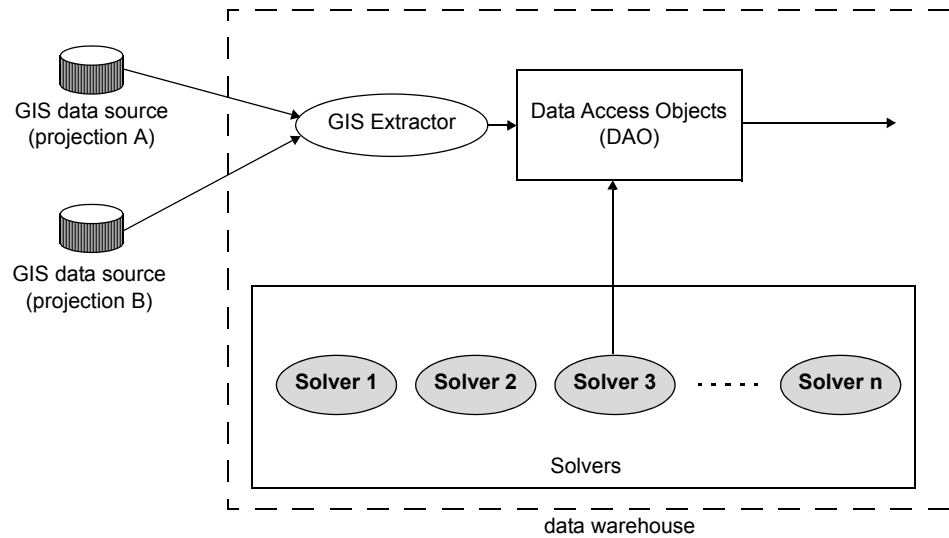


FIGURE 32. Solvers in the data warehouse

The following chapter describes a first prototype implementation of the unified object-oriented data model and the urban data warehouse.

CHAPTER 9

Prototype Implementation and Validation

This chapter describes a first prototype implementation of the unified object-oriented data model described in Chapter 7 and the urban data warehouse described in Chapter 6 and Chapter 8. It ends by summarizing the tests that were performed to validate the underlying approach.

9.1 Assumptions

The purpose of this prototype is to test both the proposed data warehouse architecture and the unified object-oriented data model with real data in a realistic city (Makkah). This was done by simulating existing heterogeneous data sources available in Makkah and

simulating heterogeneous applications currently used by urban designers in the city.

For administrative and technical reasons, this prototype could not communicate directly with the real data sources. A representative selection of data sources was *simulated* with reasonable equivalents and populated with real data. All the available data sources fall into one of the following categories: DBMS, CAD, and GIS. To simulate a DBMS source, a reasonable proxy for an existing DBMS source was implemented and populated with real data obtained from that source *using the same schema*. To simulate a CAD source, a copy of a CAD file was obtained from an existing CAD file used as source. To simulate a GIS source, a reasonable proxy for an existing GIS source was implemented and again populated with real data obtained from a that source. The simulated source uses the same schema and software.

Some of the applications currently used by urban designers in Makkah were also simulated with reasonable equivalents. It so happens that the software platforms commonly used by urban designers in Makkah are identical to the platforms used to simulate the data sources. But, in each simulated application, the needed data come from at least two of the simulated heterogeneous sources. The data integrated into the unified data model were then translated into the application schema.

This prototype aims to test how correctly and conveniently the warehouse, for the context in Makkah, is able to extract and transfer data to the (simulated) applications. The assumption is that if the

validity of the approach can be demonstrated for a city like Makkah, the approach is generalizable and valid for a broad range of other cities.

9.2 Software Platform

As stated in Section 5.3.4, several criteria should be used to choose an efficient software platform to implement the data warehouse and the data model. Based on these criteria, the Java programming language and the Extensible Markup Language (XML) were used for a first prototype. Java provides portable code since it is system-independent, and XML provides portable data since it is application-independent. Given the heterogeneity of the data sources at one end and the applications at the other end, using both portable code and portable data provides an appropriate development environment for implementing the data warehouse. In addition, Java supports object-oriented features that are needed to implement the unified object-oriented data model. Java also supports building web-enabled applications. This is important since the urban data warehouse is to be web-enabled. XML provides an open-standard file format that can be used by different urban design applications. Given an XML file, an urban designer can easily use or build a data translator to insert the requested data into various applications.

In addition, a web server has been set up to implement the proposed web-enabled data warehouse architecture. Microsoft's IIS (Internet

Information Services) was used to install the web server. IIS is a software component that turns a PC into a web server. Jakarta Tomcat (from Apache Software Foundation) has been installed in the web server to allow running Java Servlets. A Servlet is a Java class or code that runs on the server side upon receiving client HTTP requests. They are able to respond to HTML form requests. Java Servlets were used in this prototype to implement DAOs that receive HTTP requests from the web interface.

9.3 Simulating Heterogeneous Data Sources

For the reasons mentioned above, this first prototype could not be exercised directly on the data sources owned by various local authorities in the city of Makkah. To be nevertheless able to experiment with the various types of heterogeneous data sources commonly used in cities such as Makkah, this prototype simulates existing sources by reasonable equivalents and populates the sources with sample data obtained from the real sources.

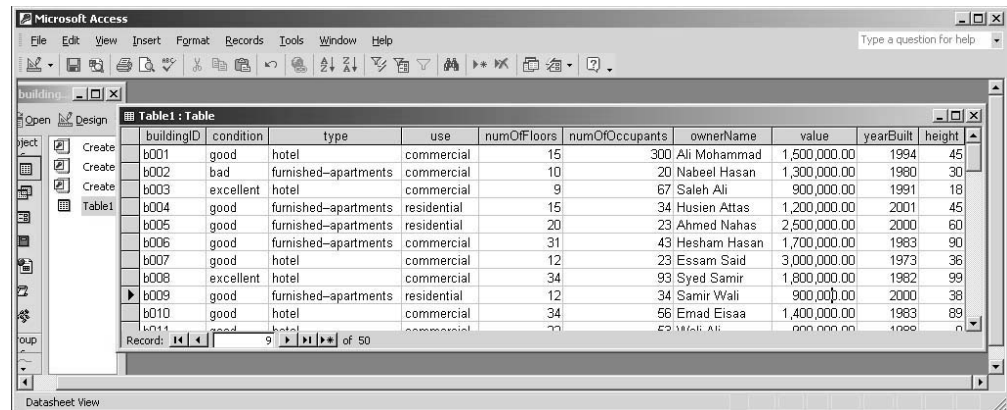
9.3.1 Relational Database Management System

A DBMS (Relational Database Management System) data source was simulated using MS Access. This data source simulates the relational database system available at the Ministry of Hajj described in

Section 3.3. The DBMS at the Ministry of Hajj is implemented using Oracle. The system provides information about buildings available for pilgrims coming to Makkah each year. The building data available in the system include building ID number, condition, type, use, number of floors, number of occupants, owner name, value, year built, and height. The schema used to store these data is a flat table. Each row (record) in the table represents a building and captures all its non-geometric attributes in columns. Each column in the table represents a single attribute like “use”. A value is stored in the table for each building attribute. For instance, a building use attribute may have different values like “commercial”, “residential”, or “mixed” according to the use of that particular building in that row.

The MS Access data source, shown in Figure 33, simulates the Ministry of Hajj data source in terms of data and schema. MS Access is a relational DBMS like Oracle. The main difference between Oracle and Access is that Oracle is designed for big corporate applications, while Access is designed for personal or small business applications. Oracle is more reliable than Access when the database includes a large number of records. Both Oracle and Access support SQL (Structured Query Language) as a language for creating, manipulating, and querying the database. For the purpose of this prototype, Access appears to be sufficient since only sample data are used.

A table is created in MS Access to simulate the Oracle table available at the Ministry (see Figure 33). The columns capture the different attributes describing a building as discussed above.



buildingID	condition	type	use	numOfFloors	numOfOccupants	ownerName	value	yearBuilt	height
b001	good	hotel	commercial	15	300	Ali Mohammad	1,500,000.00	1994	45
b002	bad	furnished-apartments	commercial	10	20	Nabeel Hasan	1,300,000.00	1980	30
b003	excellent	hotel	commercial	9	67	Saleh Ali	900,000.00	1991	18
b004	good	furnished-apartments	residential	15	34	Husien Attas	1,200,000.00	2001	45
b005	good	furnished-apartments	residential	20	23	Ahmed Nahas	2,500,000.00	2000	60
b006	good	furnished-apartments	commercial	31	43	Hesham Hasan	1,700,000.00	1983	90
b007	good	hotel	commercial	12	23	Essam Said	3,000,000.00	1973	36
b008	excellent	hotel	commercial	34	93	Syed Samir	1,800,000.00	1982	99
b009	good	furnished-apartments	residential	12	34	Samir Wali	900,000.00	2000	38
b010	good	hotel	commercial	34	56	Emad Eisaa	1,400,000.00	1983	89

FIGURE 33. Simulated DBMS (MS Access) data source with sample data

9.3.2 Computer-Aided Design System

A CAD (Computer Aided Design) data source was simulated using AutoCAD by Autodesk. This data source simulates the AutoCAD files available at the Ministry of Defense described in Section 3.1. The Ministry of Defense uses aerial photography to generate AutoCAD maps for the city of Makkah. These maps are indexed geographically to cover the whole city. The city is divided into adjacent rectangular maps, and a key map can be used to select the area for which a more detailed map should be retrieved. The detailed map offers an accurate source of

2D geometries of different urban objects in the city. Each map is available in AutoCAD's DXF file format and includes layers that represent geometries of buildings, streets, land parcels, blocks, contour lines, elevation points, street trees and lights, street names, and landscape features.

To simulate this data source, one of the map files was selected and a copy obtained from the Ministry of Defense as a DXF file. The selected map, shown in Figure 34, represents an urban area located in central Makkah. Some extra work needed to be done for this data source, since CAD files contain collections of geometries without any keys (reference attributes) as used in GIS and DBMS. For this reason, geometric entities that describe a specific type of object or instance cannot be selected from the CAD file directly. To resolve this issue, a coding technique was used to tag each urban object with a key (ID). A building, for instance, is given a key number like B034. All geometric entities that belong to that building are grouped and tagged with this key number.

Based on these tags, geometric information about different urban objects can be identified and may then be extracted from the DXF file. Note that if a data warehouse as it is envisioned here needs to retrieve data from a traditional CAD file, the warehouse administrator has to add appropriate tags to all CAD files that do not have them.

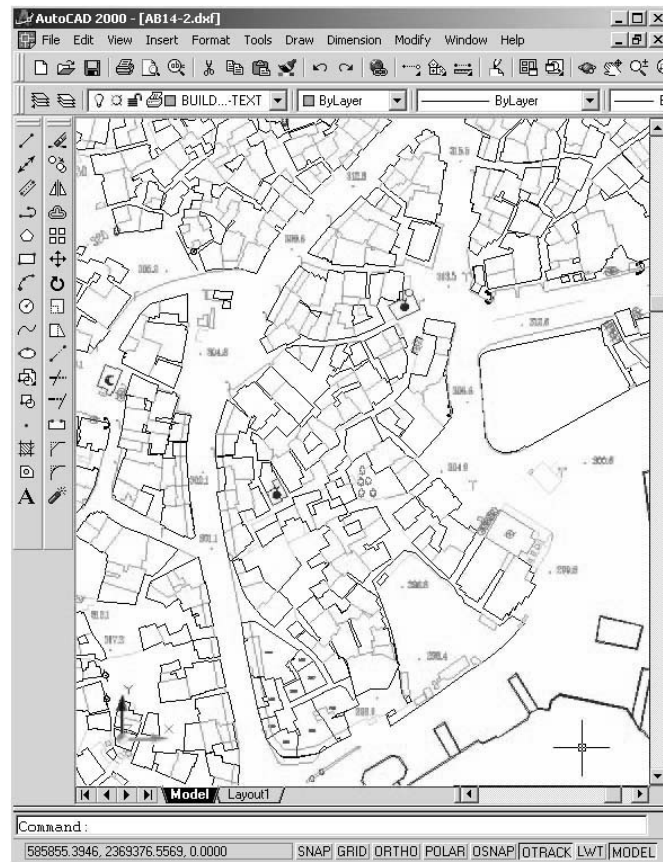


FIGURE 34. Simulated CAD (AutoCAD) data source with sample data

9.3.3 Geographic Information System

A GIS (Geographic Information System) data source was simulated using ArcView GIS software by ESRI. This data source is a direct proxy for the ArcView GIS available at The Custodian of the Two Holy

Mosques Institute for Hajj Research described in Section 3.4. The GIS system at the Institute stores and manages information about accommodation buildings (hotels and furnished-apartment buildings) available for visitors to the city. The data available about each building include building ID number, address, name, number of dwellings, number of rooms per dwelling, and building 2D geometry.

The schema used to store the non-geometric data of a building is a flat table just like a DBMS table described in Section 9.3.1. Each record (row) in the table represents a building and captures non-geometric attributes of the building. The table is stored as a binary file in DBF format. In addition to non-geometric data, the GIS stores the geometries of buildings as a GIS map using the global coordinate system (latitudes and longitudes). ArcView stores a building's geometry as a closed polygon with multiple corner points. Each point has a latitude (X) and longitude (Y) value. Geometric data are stored in a binary SHP file (shape file). To link the shape file (SHP) with the DBF file, an index (SHX) file is used. The index file stores pointers to link the non-geometric attributes to the geometric data about each building.

To simulate this data source, the same software (ArcView) was used as shown in Figure 35. The simulated data source uses the same schema as the GIS data source at The Custodian of the Two Holy Mosques Institute for Hajj Research. The urban area selected in the simulated CAD source, described in the previous section, is also used to obtain sample GIS data from the Institute.

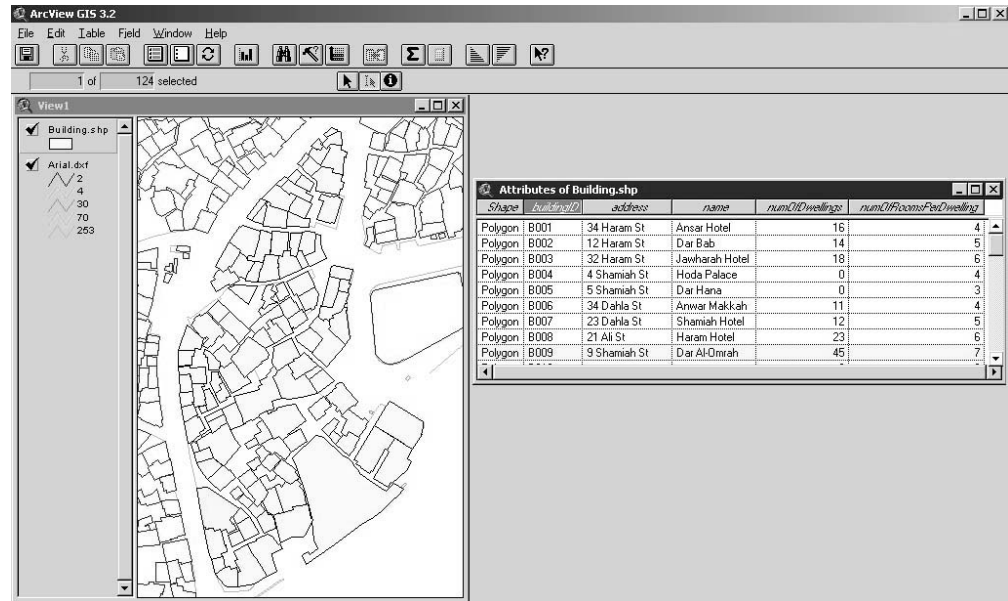


FIGURE 35. Simulated GIS (ArcView) data source with sample data

9.4 Unified Object-Oriented Data Model

The unified object-oriented data model, described in Chapter 7, was implemented using Java classes and added to the simulated web server. Each class represents the data structure for a type of urban object and captures all non-geometric attributes *and* geometries of objects of that type. When the system is running, urban (domain) objects are created as instances of the corresponding Java classes.

Each urban object, a Building object for instance, is used by a corresponding DAO to capture the retrieved data from heterogeneous sources.

As an example, Figure 36 shows Java code that implements the Building class described in Section 7.3.5. The Building class is associated with the BuildingGeometry class. The BuildingGeometry class is also implemented using Java classes as well as all other classes that describe building geometries, including Solid, Prism, Polygon3D, Point3D, ExtrudedSolid, ClosedShape2D, Polygon2D, Ellipse, Circle, and Point2D.

```
public class Building {  
  
    String buildingID = new String();  
    String address = new String();  
    String use = new String();  
    String name = new String();  
    String type = new String();  
    String condition = new String();  
    String owner = new String();  
    String materials = new String();  
    int numberOfFloors;  
    int numberOfOccupants;  
    String character = new String();  
    double value;  
    String accessibility = new String();  
    String architect = new String();  
    int yearBuilt;  
    int numberOfDwellings;  
    int numberOfRoomsPerDwelling;  
    BuildingGeometry geometry = new BuildingGeometry();  
  
}
```

FIGURE 36. Building class definition in Java code

9.5 Data Access Objects

For each class defined as part of the unified object model, a DAO must be implemented. DAOs (described in Section 8.1) are implemented using Java Servlets and added to the simulated web server. A Servlet is a Java class that receives HTTP requests and then posts data to a web browser or an application. When the system is running, a DAO object is associated with a corresponding urban (domain) object from the unified object-oriented data model. A DAO uses this domain object to capture the retrieved data coming from heterogeneous sources. For example, a *BuildingDAO* object uses a *Building* object from the unified model to integrate and capture the data retrieved from different data sources.

As an example, Figure 37 shows a partial definition of the *BuildingDAO* class in Java code. The *BuildingDAO* takes a global identification number (ID) as an argument and creates an instance of the corresponding *Building* object. It then assigns a value for each attribute using a get method of an appropriate extractor. Each get method takes a building ID, an attribute name, and a reference URL as arguments. It connects to the appropriate data source and returns a value for the attribute. All retrieved values are captured by the *Building* object.

```
public class BuildingDAO extends HttpServlet {
    protected void doPost (HttpServletRequest request, HttpServletResponse response)
    {

        // create instance of a building object
        Building building = new Building();
        building.buildingID = buildingID;

        // assign values for each attribute using a get method (operation)
        building.address = getAddress();
        building.use = getUse();
        building.name = getName();
        building.type = getType();
        building.condition = getCondition();
        building.owner = getOwner();
        building.materials = get();
        building.numberOfFloors = getNumberOfFloors();
        building.numberOfOccupants = getNumberOfOccupants();
        building.character = getCharacter();
        building.value = getValue();
        building.accessibility = getAccessibility();
        building.architect = getArchitect();
        building.yearBuilt = getYearBuilt();
        building.numberOfDwellings = getNumberOfDwellings();
        building.numberOfRoomsPerDwelling = getNumberOfRoomsPerDwelling();
        building.buildingGeometry = getBuildingGeometry();

        // example get operation (method) to get building use data
        public static getUse() {
            String dataSourceULR = new String("dataSourceDBMS");
            String use = new String();
            use = GISextractor(dataSourceULR, SQLstatemnet);
            return use;
        }
        ...
        ...
    }
}
```

FIGURE 37. BuildingDAO class definition in Java code

9.6 Extractors

Three kinds of data extractors have been implemented as Java classes, one for each of the simulated data sources described in Section 9.3.

9.6.1 DBMS Extractor

In this prototype, a DBMS Extractor has been implemented to extract data from the MS Access data source described in Section 9.3.1. It receives a data request from a DAO, connects to the MS Access database, extracts the requested data, and then returns these data to the DAO. The DBMS extractor takes two arguments: a URL (Uniform Resource Locator), and an SQL (Structured Query Language) statement. The URL indicates where the local or remote MS Access database is located. The SQL statement specifies the selection criteria of the requested data. The DBMS extractor needs these arguments to connect to the database and retrieve the requested data.

In this implementation, Microsoft's Open Database Connectivity (ODBC) technology was used to allow querying the MS Access database through the data warehouse. ODBC is a software component that comes with MS Windows and allows direct interaction with a local or remote database without the need to run MS Access directly. Java, in turn, has a software component called "driver" to allow communicating with ODBC. The driver needs a URL and an SQL statement to obtain

data from the database. The DBMS Extractor uses this driver to query the MS Access database and then receives a result data set from the database. The Extractor then returns this data set to the DAO.

9.6.2 CAD Extractor

A CAD Extractor has been implemented to extract data from the simulated AutoCAD data source discussed in Section 9.3.2. The CAD Extractor receives a data request, locates a DXF file, and then parses and extracts the requested geometric data from the file. This extractor takes a URL and an object ID number as an argument. The URL indicates where the DXF file is located. The object ID specifies for which urban object, a building for instance, geometric information should be obtained. The CAD Extractor uses these arguments to connect to the specified DXF file, extract the object geometry, and return the requested data to the DAO.

The DXF file stores descriptions of different geometries available in that file. A polygon, for instance, is stored as a collection of corner points of X and Y values. To find a polygon in the DXF file, the tags described in Section 9.3.2 have been used. The CAD Extractor uses a tag (key number) to find a polygon (representing a building) and then extracts its data (corner points).

9.6.3 GIS Extractor

A GIS Extractor has been implemented to work with the ArcView GIS data source described in Section 9.3.3. The GIS Extractor receives a data request, connects to a GIS data source, and then returns the requested data to a DAO. This extractor takes a URL and an SQL statement as arguments. The URL indicates where the GIS data source is located, and the SQL statement specifies the selection criteria of the requested data. The GIS Extractor then connects to the specified data source and returns the requested data to the appropriate DAO.

As described in Section 9.3.3, ArcView uses three files (SHP, SHX, and DBF) to capture geometric and non-geometric data about a particular urban object, like a building. Each building geometry is stored in the binary SHP file as a polygon that consists of multiple corner points. Using a building ID number, the GIS Extractor finds its corresponding polygon in the binary SHP file and then extracts its geometric data (corner points).

In addition, the GIS Extractor uses the SQL statement to query the DBF file that contains the non-geometric attributes about the same building. Again, ODBS technology described above was used to communicate directly with the DBF file.

9.7 Solvers

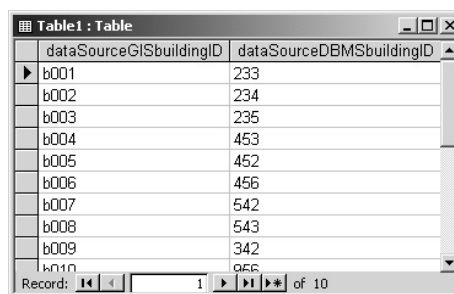
A sample Solver has been implemented to resolve a typical data integration problem: the problem that arises when the key attribute for the same object is different at two different data sources, for example, the same building has ID b001 at the DBMS data source and ID 233 at a the GIS data source (see Figure 38). A solver handling this problem maps the building identification numbers between the two data sources. The Solver was implemented using Java and then added to the data warehouse as a software component that is called by a DAO whenever this integration problem occurs.

DBMS Data Source		GIS Data Source	
buildingID	address	buildingID	use
b001	34 Haram St.	233	commercial
b002	12 Haram St.	234	commercial
b003	32 Haram St.	235	residential

FIGURE 38. Different key attributes (IDs) problem

The solver uses a lookup table to find a key attribute in the MS Access data source and to get its corresponding key attribute in the ArcView data source. The table consists of two columns as shown in Figure 39. In each row of that table, the first column has a key attribute for a building in the MS Access data source, and the second column has a corresponding key attribute for the same building at the ArcView data

source. The Solver has a method named Solve which takes a building ID number as an argument, finds the building ID number at the first data source, and then returns its corresponding building ID number at the second data source. The DAO then uses the right building ID to get its data using Extractors.



dataSourceGISbuildingID	dataSourceDBMSbuildingID
b001	233
b002	234
b003	235
b004	453
b005	452
b006	456
b007	542
b008	543
b009	342
b010	956

FIGURE 39. Lookup table used by the Solver

A more general model for solvers is beyond the scope of this dissertation (see Section 10.2.2).

9.8 Client Interfaces

Two kinds of client interfaces have been implemented to communicate with the data warehouse. The first interface works through a web browser that can be used to select, retrieve, and inspect data. The second one uses an application API (Application Programming Interface) to communicate directly with the data warehouse without user interaction.

9.8.1 Web Interface

An urban designer can use the web interface to retrieve and inspect data of interest. This interface is developed using HTML (Hyper Text Markup Language) pages, which allows the user to specify the data of interest. Figure 40 shows the first page of the web interface, which allows the user to choose the type of urban objects for which data should be obtained. Figure 41 shows the next web page where the user can select the needed attributes for the selected type, in this case a Building object.

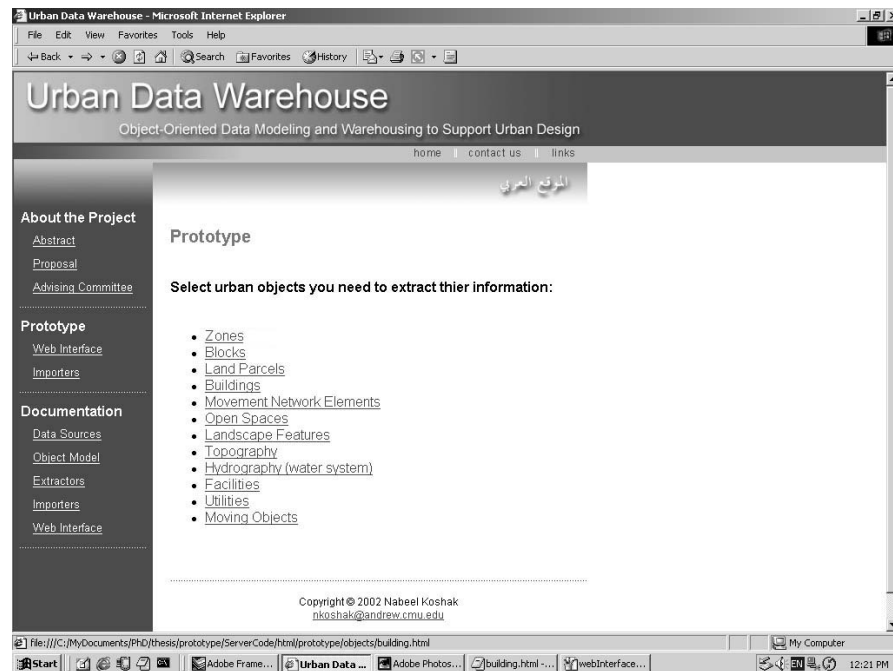


FIGURE 40. Web Interface to the Urban Data Warehouse, showing a list of urban object types

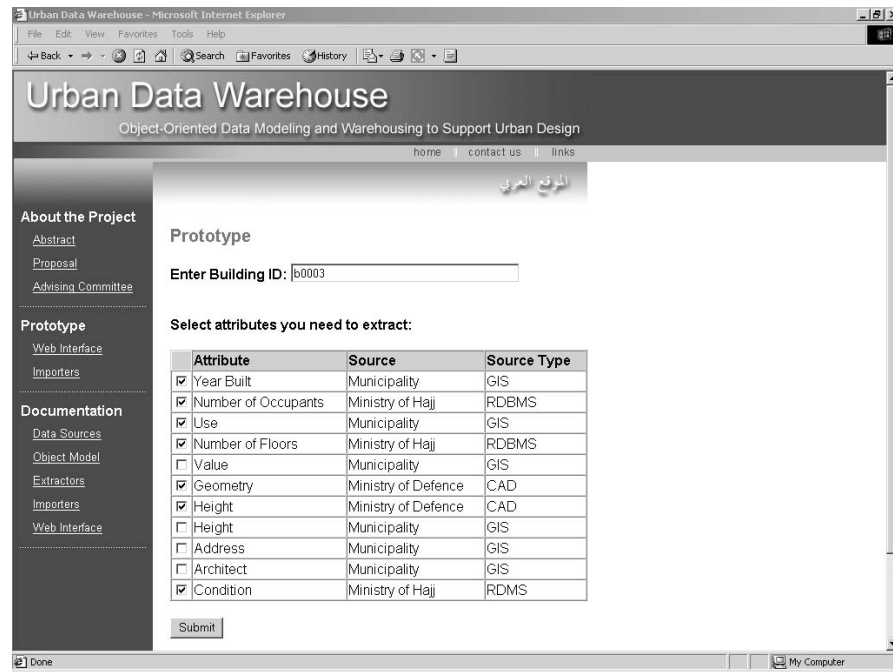
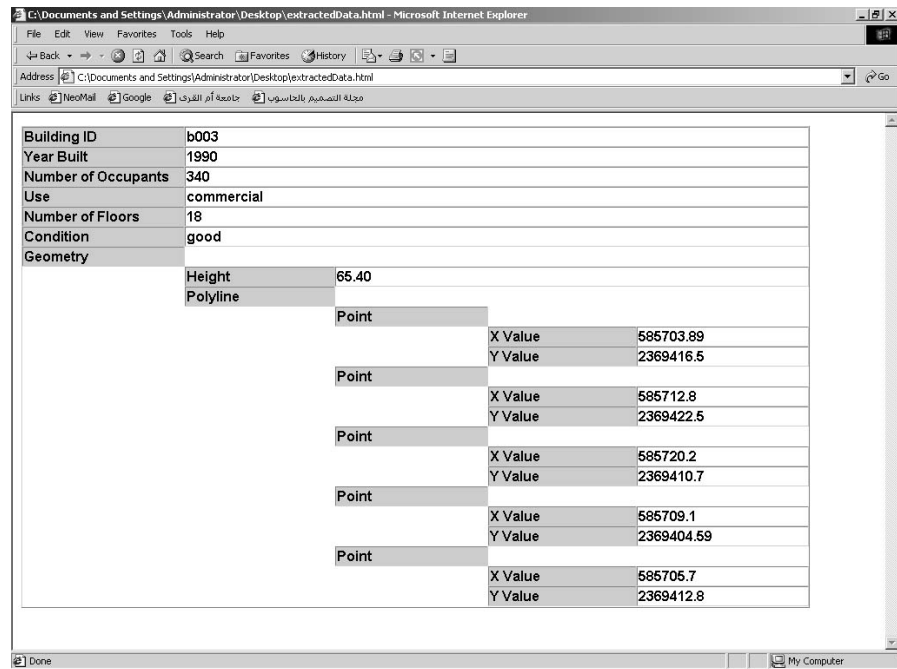


FIGURE 41. Web Interface, showing a list of available attributes for a Building object

When the user clicks on the “Submit” button in the web interface, an HTTP (Hyper Text Transfer Protocol) request is sent to the simulated web server. A corresponding DAO available on that server receives this data request and communicates with other components of the data warehouse to retrieve the requested data. The DAO then returns the requested data to the user through the web interface and presents the retrieved data as HTML tables (see Figure 42). The user can then use the web browser to view the retrieved data.



The screenshot shows a Microsoft Internet Explorer browser window with the address bar displaying 'C:\Documents and Settings\Administrator\Desktop\extractedData.html'. The main content area displays an HTML table with the following data:

Building ID	b003
Year Built	1990
Number of Occupants	340
Use	commercial
Number of Floors	18
Condition	good
Geometry	
Height	65.40
Polyline	
Point	
X Value	595703.89
Y Value	2369416.5
Point	
X Value	585712.8
Y Value	2369422.5
Point	
X Value	585720.2
Y Value	2369410.7
Point	
X Value	585709.1
Y Value	2369404.59
Point	
X Value	585705.7
Y Value	2369412.8

FIGURE 42. Sample extracted data represented as HTML Tables

The data can also be downloaded to the user's machine as an XML file. The XML file (see Figure 43) can then be parsed and used to import the retrieved data into different applications, as will be demonstrated in Section 9.9.

Note that it would have been more elegant (and simpler) to implement DAOs that always return data in the form of XML files and let the client worry about what to do with them. But this would have forced every client to create a parser and interface to some application. This is

avoided by enabling the data warehouse to return data as web pages; this can be considered an extra service offered by the warehouse.

```
<?xml version="1.0"?>
<building>
  <buildingID>b003</buildingID>
  <yearBuilt>1990</yearBuilt>
  <numberOfOccupants>340</numberOfOccupants>
  <use>commercial</use>
  <numberOfFloors>18</numberOfFloors>
  <condition>good</condition>
  <geometryGeometry>
    <height>65.40</height>
    <polygon2D>
      <point2D>
        <xValue>585703.89</xValue>
        <yValue>2369416.5</yValue>
      </point2D>
      <point2D>
        <xValue>585712.8</xValue>
        <yValue>2369422.5</yValue>
      </point2D>
      <point2D>
        <xValue>585720.2</xValue>
        <yValue>2369410.7</yValue>
      </point2D>
      <point2D>
        <xValue>585709.1</xValue>
        <yValue>2369404.59</yValue>
      </point2D>
      <point2D>
        <xValue>585705.7</xValue>
        <yValue>2369412.8</yValue>
      </point2D>
    </polygon2D>
  </buildingGeometry>
</building>
```

FIGURE 43. The content of the XML file that includes the sample extracted data

9.8.2 Application Interfaces

Another option for interacting with the data warehouse is to use an Application Programming Interface (API) to communicate directly with the warehouse in order to retrieve the needed data. This interface does not require a web browser because the needed data is predefined and the API connects directly to the data warehouse using DAOs. Since DAOs have been implemented using Java Servlets, an application can communicate with these Servlets using HTTP requests. Through the API, the retrieved data is directly imported into an application without user interruption. Section 9.9.1 will show example data extracted and imported into an application using an API. In this example, the application connects directly to the data warehouse and imports the retrieved data into a DBMS application.

9.9 Heterogeneous Applications

The software used to simulate three useful urban design applications is the same as the software used to simulate the data sources. These software packages are commonly used by different urban designers and planners in the city of Makkah. But note that these applications do *not* use the same data as the data sources implemented through the

same software. Each application needs data from at least two sources, obtains them from the warehouse, and integrates them as required.

It was also easier to simulate applications with the typical data needs of urban design using software that the author of this dissertation already understood, rather than having to write the applications from scratch or having to familiarize himself with yet another piece of software.

9.9.1 Housing Stock Reporter

This application simulates a housing stock reporter that generates a statistical report about housing stock for a specified urban area. This application is used by local authorities in Makkah to investigate housing supply and demand for accommodating the huge number of pilgrims. For a selected urban area, the application generates a report that includes the number of buildings, total number of occupants, total number of dwellings, and average number of rooms per dwelling. This application is simulated using MS Access. Java code has been written to connect directly to the DAOs, to retrieve the requested data as an XML file, and then to parse the file and import the data into a table in Access. Based on that table, MS Access can then automatically generate housing stock reports about the selected urban area.

Figure 44 shows a report generated by the application using data extracted from the data warehouse.

This housing stock reporter needs data from two heterogeneous data sources: the simulated GIS (ArcView), which provides building data, including number of dwellings, and number of rooms per dwelling; and the simulated DBMS, which provides other building data, including number of floors and number of occupants.

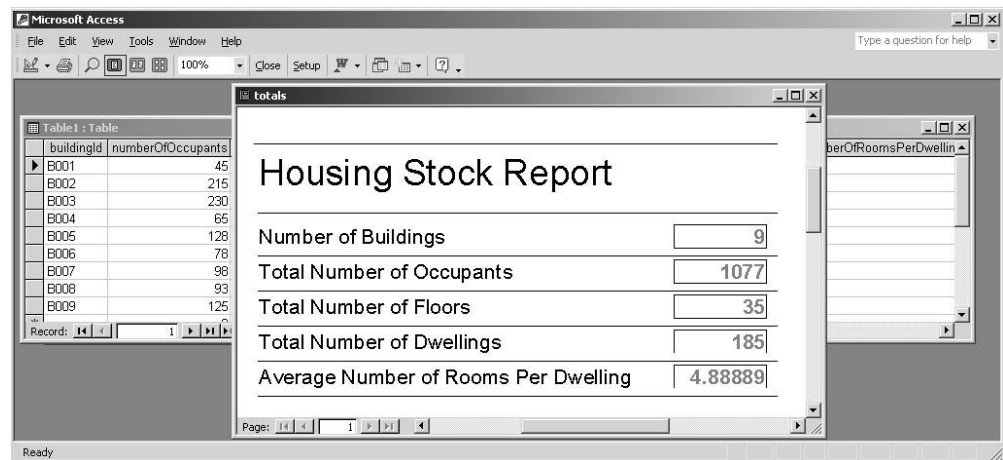


FIGURE 44. Housing stock reports generated using the imported XML data

9.9.2 3D Modeler and Visualizer

This application simulates a 3D modeler and visualizer used by The Custodian of the Two Holly Mosques Institute for Hajj Research to show

in 3 dimensions existing and proposed buildings, where the existing buildings are modeled based on data received from a GIS and a DBMS. Proposed buildings are modeled locally in the same modeler so that their form and impact on the neighborhood can be assessed.

Figure 45 illustrates the level of detail at which building geometries are currently available from the authorities in Makkah. They are given by the outline of their footprints and their height. From these data, an application like the present one is able to construct 3 dimensional building geometries as extruded solids. The building geometry representation in the unified object model goes beyond this, as Figure 21 demonstrates. This illustrates how the object model anticipates some data needs of potential applications that cannot be satisfied by the data sources currently maintained in Makkah. That is, the model is more general than the context of Makkah requires.

AutoCAD is used to simulate the application. AutoCAD's API, Visual Basic for Application (VBA), is used to parse an XML file returned by the data warehouse and import the data into AutoCAD. A translator has been implemented using AutoCAD's VBA to parse the XML file, construct a 3D model of the buildings, and draw dynamically a 3D representation of these buildings on the screen. Figure 45 shows the results of a test run of the application focusing on a block in central Makkah.

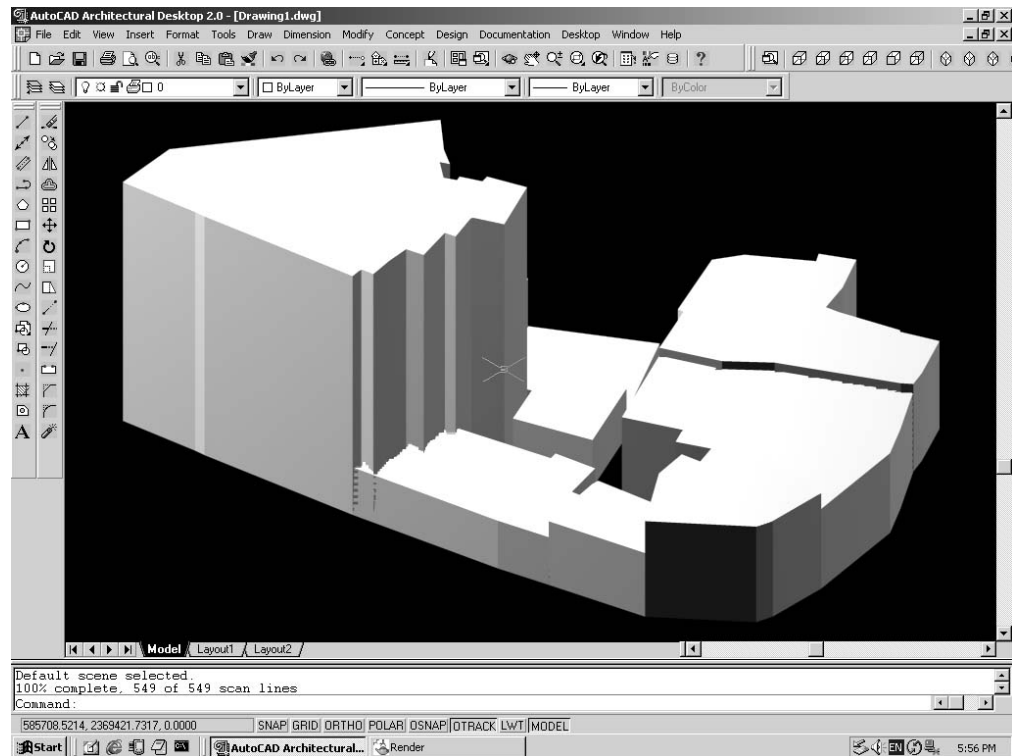


FIGURE 45. 3D Visualization of building data extracted from XML data

The visualizer needs data from two heterogeneous data sources, the simulated GIS (ArcView), which provides building geometry data in the form of a 2D polygon representing the footprint of each building; and the simulated DBMS (MS Access) data source, which provides building height information.

9.9.3 Map Generator

The third application is an urban map generator used by The Custodian of the Two Holly Mosques Institute for Hajj Research to create maps of an urban area with different color coding according to building height, year built, number of occupants, and use. These maps can then be used by urban designers to visually analyze changes in an urban environment. For example, one of these maps can be used to visualize land use patterns. Figure 46 shows the application in a test run after the data have been extracted from an XML file and imported.

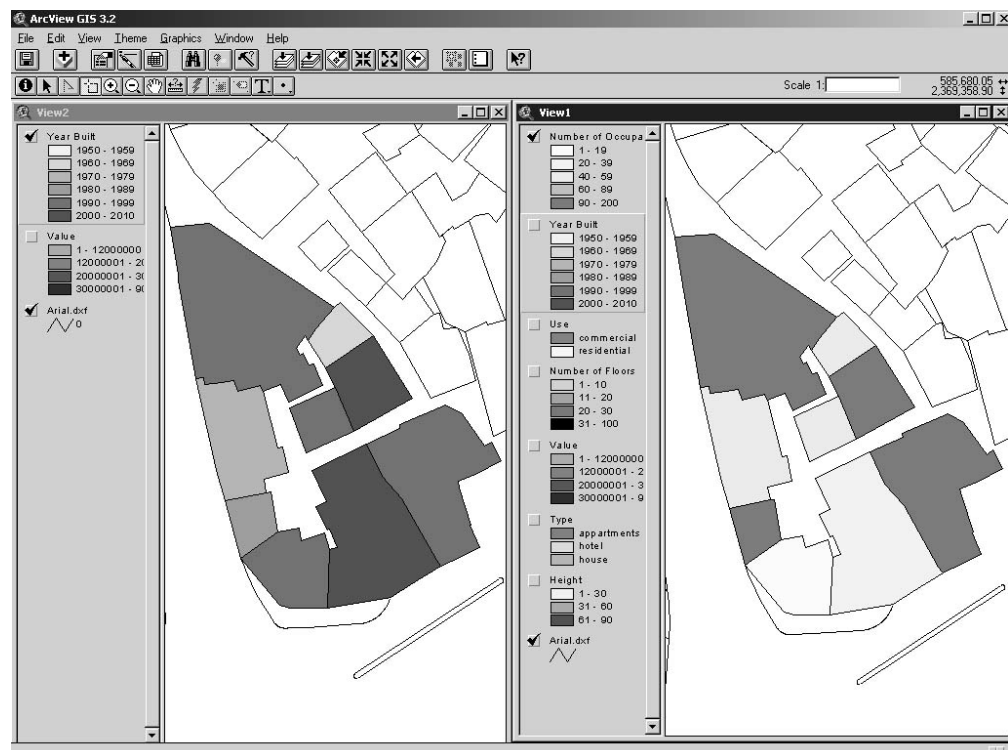


FIGURE 46. Maps generated from XML data

This application was simulated using again ArcView. It needs data from all three heterogeneous data sources: the simulated AutoCAD data source to provide a building's geometry, which is a 2D polygon representation of that building; the simulated MS Access data source that provides other building information, including number of floors, value, type, number of occupants, year built, and condition; and the simulated GIS data source that provides building name, address, number of dwellings, and number of rooms per dwelling. Data coming from the CAD, DBMS, and GIS data sources are integrated into the XML file used in this example application.

9.10 Results

This chapter has demonstrated a working implementation of the proposed urban data warehouse architecture. This implementation shows how all components of the warehouse, including the DAOs, Extractors, Solvers, and unified schema work together to satisfy the crucial requirements of the data warehouse. The prototype was tested on heterogeneous data sources simulating existing data sources in Makkah rather faithfully and heterogeneous applications whose functionalities and data needs resemble closely those of applications

used currently by urban designers in Makkah. Test runs of the applications validated the prototype and the underlying assumptions.

This prototype has also demonstrated that the proposed object-oriented data model is able to capture and integrate heterogeneous data as needed by the applications. It is also able to provide urban designers with unified data in a standard file format (XML) for use by heterogeneous urban design applications.

In general, this prototype shows that the proposed warehouse architecture and the model will cover the situation in the city of Makkah. The expectation is that the architecture and model are general and can be implemented by cities other than Makkah.

CHAPTER 10

Conclusions

This chapter states contributions of this dissertation and discusses possible future research issues.

10.1 Contributions

This research makes the following contributions to the field of computer-supported urban design.

10.1.1 Solution to a Very Pressing Problem in Computer-Supported Urban Design

This dissertation has provided a solution to a central problem in computer-supported urban design and planning: the heterogeneity of

data sources needed by clients such as urban designers and the applications they use. If urban designers and planners want to use software that needs input from these heterogeneous sources, they must transform these data manually into the required format, a time-consuming and error-prone task that may, in fact, preclude using the software at all, thus depriving designers of possible crucial design support tools. This dissertation introduces a computational environment that hides the heterogeneity of data sources and provides urban designers with a unified data format that can be used by heterogeneous urban design applications.

The survey in Chapter 2 demonstrates that this is the first project, to the best of the author's knowledge, to address the general data integration problem in urban design and to provide a viable solution to this problem.

10.1.2 Urban Data Warehouse and Object Model

Specifically, this dissertation has demonstrated how *data warehousing*—combined with *object-oriented data modeling*—is able to provide a general solution for the problem of heterogeneity of data sources. An urban data warehouse provides an intermediate layer

between the sources and clients that isolates each side from changes and modifications on the other side.

This research has presented an architecture for the urban data warehouse that is general and can be implemented by cities and municipalities based on commonly available platforms and software. The architecture has been tested for the city of Makkah, but it is also possible to implement it for other cities facing the same problem.

The simple object model (schema) developed for this dissertation can be used to provide data to clients in a unified format that is independent from the data sources themselves. The model has the following features:

- The model is comprehensive within the scope of this dissertation, since it captures the content of the various forms of digital representations of urban areas currently available in the city of Makkah. In selected areas, the model exceeds what is currently available in Makkah.
- The model is, on the other hand, simple enough to facilitate parsing and interpretation by clients receiving data based on this model.
- The model is extensible so that a data warehouse administrator is able to modify existing objects or add new objects to the model. This facilitates accommodating future requirements of the

data warehouse and enables modifications of the model to meet special data needs for certain cities.

The warehouse is able to use the model to generate user and application interfaces for making the objects of the model available across various current systems.

10.1.3 Portable Model, Architecture, and Implementation

The dissertation has also demonstrated that the object model and warehouse architecture can be implemented using commonly available platform-independent languages and tools. The city of Makkah has been used as a test case, but once the model has been validated, any city within Saudi Arabia (like Madinah, Riyadh, or Jeddah), even cities outside the country, can implement the same approach to support that city's urban design applications. Software that implements the DAOs can be directly shared between such cities.

10.1.4 Supporting Collaboration in Urban Design

The data warehouse is able to contribute to a better environment for urban research and development in cities such as Makkah. It establishes a collaborative environment for the various authorities influencing urban design, planning, and management and for the designers and planners involved in urban decision-making. For

instance, the data warehouse can be used in Makkah by more than one local authority (data source) at the same time to perform various tasks for the same urban area. For example, The Municipality of Makkah can work on updating geometry changes in an urban area; The Custodian of the Two Holy Mosques Institute for Hajj Research can work on extracting some transportation data to be used for a traffic simulation application; and The Ministry of Hajj can work on updating some housing data. These three organizations can simultaneously perform these different tasks using the unified integrated communication framework established by the model and the warehouse.

10.2 Future Research Issues

The work described here suggests several directions for future research. This section provides a brief outline of such issues.

10.2.1 Geometry Representation

The most important immediate issue that has to be addressed in the context of the work presented here is how to generally represent the geometry of various urban objects. The representations of object geometries in the unified object model were developed on in ad-hoc fashion in response to data available/needed for the city of Makkah. However, the issue requires a more detailed study that aims at finding a

consistent representation for object geometries that is flexible and general enough to cover all important cases across a broad spectrum of needs and applications, while remaining simple enough for easy parsing and interpretation.

10.2.2 Connected Community

Another open issue at the time of writing is how to construct and use solvers in general. Currently, solvers are built and added to the warehouse by its administrators. However, this need not be the case. The author envisions a situation in which the data warehouse becomes the backbone of a connected community of data providers and data users that cooperate and benefit from each other's work. At the most basic level, an authority maintaining one data source serving external clients may become a client in turn when it needs data maintained by other authorities. The author suggests going a step further and having data providers as well as data users share knowledge by using the data warehouse to support their design decisions. Solvers could be used to make these insights available to the entire community.

In a more effective collaborative environment, solvers can be developed by urban designers and shared—via the Internet—through the data warehouse as distributed reusable software components. The Internet has become an important medium for sharing information. Many Internet applications deliver data, but the Web remains underused for remote processing of computational objects. Bhargava et

al. have described how consumers can use, over the Internet, technologies that are located at and execute on providers' machines [Bhargava et al. 1995a/b]. One application of this approach would be the use of the Internet to share Solvers for data integration problems. Solvers can be shared based on specific parameterized models set by the data warehouse administrator.

10.2.3 Decision Support System

Further research possibilities emerge once the problem of heterogeneous data sources for urban design applications has been solved. The data available through a unified interface can now be used to support more advanced decision-making tools that *generate* data from the available data. Two such promising emerging techniques are identified below:

- **Online Analytical Processing (OLAP)**

OLAP refers to an advanced data analysis environment. OLAP tools may access the data warehouse to provide advanced multidimensional data analysis to support decision making, business modeling, and operation research activities. Multidimensional aspects of data analysis tend to view data as they relate to other data [Rob & Carlos 1997]. For instance, in the context of this research, OLAP could be used to monitor and analyze urban growth and changes in cities.

- **Data Mining**

Data Mining is a new kind of specialized decision support tool that automate the analysis of data in order to reveal unknown data characteristics, relationships, dependences, and trends. Data mining tools use the data warehouse to extract the target data set. Then they will apply data mining algorithms to obtain highly specialized information or knowledge about the data and, therefore, about the organization's operations [Rob & Carlos 1997]. For instance, data mining applied to a target data set extracted from the warehouse could be used to investigate the relationship between urban growth and traffic congestion.

Bibliography

- [Al-Yafi 1993] Al-Yafi, A. (1993). Management of Hajj Mobility Systems. Joh. Enschede Amsterdam, Holland.
- [Barhamain 1997] Barhamain, S. Y. (1997). "Facilities Planning and Management for the Large-Scale Events Industry with a Particular Reference to a Typical Mega-Event, the Hajj". Ph.D. Thesis, Department of Architecture and Building Science, University of Strathclyde, UK.
- [Barnett 1982] Barnett, J. (1982). An Introduction to Urban Design. Harper and Row Publishers Inc., New York, USA.
- [Batty et al. 1998] Batty, M., Dodge, M., Jiang, B., and Smith, A. (1998). "GIS and Urban Design". Working paper at the Center of Advanced Spatial Analysis, University College of London, UK.
- [Bhargava et al. 1995a] Bhargava, H. K., King, A. S., and McQuay, D. S. (1995). "DecisionNet: An Architecture for Modeling and Design Support over the World Wide Web". In Proceedings of the International Symposium on Decision Support Systems, Hong Kong, June.
- [Bhargava et al. 1995b] Bhargava, H. K., Krishnan, R., and Muller, R. (1995). "On Parameterized Transaction Models

- for Agents in Electronic Markets for Decision Technologies". In Proceedings of the Fifth Workshop on Information Technologies and Systems, Amsterdam, Holland.
- [Booch et al. 1999] Booch, G., Rumbaugh, J., and Jacobson, I. (1999). The Unified Modeling Language. User Guide. Addison Wesley Longman, New York, USA.
- [Danahy & Hoinkes 1995] Danahy, J. W., and Hoinkes, R. (1995). "Polytrim: Collaborative Setting for Environmental Design". In the Proceedings of Computer Aided Architectural Design (CAAD) Futures '95. The Global Design Studio, Milton Tam and Robert Tech, Eds., USA.
- [Dave & Schmitt 1994] Dave, B., and Schmitt, G. (1994). "Information Systems for Urban Analysis and Design Development". In Environment and Planning B: Planning and Design, volume 21, pp. 83-96.
- [Day 1994] Day, A. (1994). "New Tools for Urban Design". Urban Design Quarterly Journal, Issue 51, July.
- [Day et al. 1996] Day, A., Robson, J., and Bourdakos, V. (1996). "Living with a Virtual City". In Architectural Research Quarterly, no 1, vol. 2.
- [Day & Radford 1998] Day, A., and Radford, A. (1998). "An Overview of City Simulation". Proceedings of The Third Conference of CAADRIA, Osaka, Japan.
- [Gross et al. 1997] Gross, M., Parker, L., and Elliott, A. (1997). "MUD: Exploring Tradeoffs in Urban Design". In the Proceedings of Computer Aided Architectural Design (CAAD) Futures Conference '97. Munich, Germany.
- [Gant & Tita 1997] Gant, J. and Tita, G. 1997. "Geographic Information Systems". Course notes. Heinz School of Public Policy and Management. Carnegie Mellon University.
- [Hariri 1986] Hariri, M. (1986). "Housing in Central Makkah: The Influence of Hajj". Ph.D. Thesis, School of

- Architecture, University of Newcastle-upon-Tyne, Newcastle-upon-Tyne.
- [Henderson & Ahearn 1998] Henderson, D. B., and Ahearn, S. (1998). "Evolution of a Municipal Landbase from Layers to Objects". In the Proceedings of Urban and Regional Information Systems Association (URISA) Conference '98.
- [Hull & Zhou 1996] Hull, R. and Zhou, G. (1996). "A Framework for Supporting Data Integration Using the Materialized and Virtual Approaches". In the Proceedings of ACM SIGMOD '96, 481-492.
- [Jiang 1997] Jiang, B. (1997). "Urban Environments: Analysis and Visualization". Working Paper, The Centre for Advanced Spatial Analysis (CASA), University College London, UK.
- [Kennedy & Kopp 2001] Kennedy, K. and Kopp, K. (2001). Understanding Map Projections. Environmental Systems Research Institute, Inc. (ESRI). Redlands, California, USA.
- [Koshak & Gross 1998] Koshak, N., and Gross, M. (1998). "3D Modeling of Historic Makkah: Strategies for Constructing Accurate CAD Models of Historic Buildings". In the Proceedings of Computer Aided Architectural Design Research In Asia (CAADRIA) Conference '98, Osaka, Japan.
- [Liggett et al. 1995] Liggett, R., Friedman, S., and Jepson, W. (1995). "Interactive Design/Decision Making in a Virtual Urban World: Visual Simulation and GIS". In the Proceedings of the 5th Annual ESRI (Environmental Systems Research Institute) User Group Conference '95.
- [Liggett & Jepson 1995] Liggett, R., Jepson, W. (1995). "Implementing an Integrated Environment for Urban Simulation: CAD, Visualization, and GIS". In Visual Databases in Architecture: Recent Advances in Design and Decision Making. Avebury.

- [Lynch 1990] Lynch, K. (1990). *City Sense and City Design*. Edited by Banerjee, T., and Southworth, M. MIT Press, Cambridge.
- [Meyer 1988] Meyer, B. (1988). *Object-Oriented Software Construction*. Prentice Hall International Ltd., UK.
- [Naiburg & Maksimchuk 2001] Naiburg, Eric J., and Maksimchuk, Robert, A. (2001). *UML for Database Design*. Addison-Wesley.
- [Papakonstantinou et al. 1995a] Papakonstantinou, Y., Garcia-Molina, H., and Widom, J. (1995). "A Query Translation Schema for Rapid Implementation of Wrappers". In the *Proceedings of Fourth International Conference on Deductive and Object-Oriented Databases*, Singapore.
- [Papakonstantinou et al. 1995b] Papakonstantinou, Y., Garcia-Molina, H., and Widom, J. (1995). "Object Exchange Across Heterogeneous Information Sources". In the *Proceedings of Data Engineering Conference*, Taipei, Taiwan.
- [Rob & Carlos 1997] Rob, P., and Carlos, C. (1997). *Database Systems: Design, Implementation, and Management*. Third Edition. Course Technology, A division of International Thomson Publishing, USA.
- [Rumbaugh et al. 1991] Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F., and Lorensen, W. (1991). *Object-Oriented Modeling and Design*. Prentice-Hall, Englewood Cliffs, New Jersey, UK.
- [Shiffer 1995] Shiffer, M. (1995). "Interactive Multimedia Planning Support: Moving from Stand-Alone Systems to the World Wide Web". *Environment and Planning B: Planning and Design*. Volume 22, pp. 649-664.
- [Shirvani 1985] Shirvani, H. (1985). *The Urban Design Process*. Van Nostrand Reinhold Company, New York, USA.

- [Silberschatz et al. 1999] Silberschatz, A., Korth, H., and Sudarshan, S. (1999). Database System Concepts. McGraw-Hill series in computer science, USA.
- [Singh 1996] Singh, R. (1996). "Exploiting GIS for Sketch Planning". In the Proceedings of Environmental Systems Research Institute (ESRI) User Conference '96, USA.
- [Smith & Dodge 1997] Smith, A. and Dodge, M. (1997). "The World Wide Web--Not just for Nerds!". Planning, pp. 16-17, Issue 1213, April.
- [Sola-Morales 2000] Sola-Morales, Pau. (2000). "Representation in Architecture: A Data Model for Computer-Aided Architectural Design". Master Thesis. Harvard University, Massachusetts, USA.
- [Wheeler & Wheeler 1998] Wheeler, J. and Wheeler, W. (1998). "Design Patterns for Persistence Independence". Java Pro Magazine, November 2001, Volume 5, Number 11. Fawcette Technical Publications, USA.
- [Yin & Williams 1995] Yin, Z., and Williams, T. H. L. (1995). "GIS Analysis Model for Urban Strategic Planning". In the Proceedings of International Symposium on RS, GIS, and GPS, Hong Kong.
- [Zeiler 1999] Zeiler, M. (1999). Modeling Our World: The ESRI Guide to Geodatabase Design. Published by Environmental Systems Research Institute, Inc. (ESRI), Redlands, California, USA.

Appendix

UML Notation

This appendix explains the Unified Modeling Language (UML) notation used in Chapter 7 to illustrate the unified object-oriented data model [Booch et al. 1999].

