

A CAD-BASED MODEL FOR SITE LAYOUT

Farnaz Sadeghpour

A Thesis

in

The Department

of

Building, Civil, and Environmental Engineering

Presented in Partial Fulfillment of Requirements
for the Degree of Doctor of Philosophy at
Concordia University
Montreal, Quebec, Canada

October 2004

© Farnaz Sadeghpour



Library and
Archives Canada

Bibliothèque et
Archives Canada

Published Heritage
Branch

Direction du
Patrimoine de l'édition

395 Wellington Street
Ottawa ON K1A 0N4
Canada

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

ISBN: 0-612-96953-3

Our file Notre référence

ISBN: 0-612-96953-3

The author has granted a non-exclusive license allowing the Library and Archives Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque et Archives Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

In compliance with the Canadian Privacy Act some supporting forms may have been removed from this thesis.

Conformément à la loi canadienne sur la protection de la vie privée, quelques formulaires secondaires ont été enlevés de cette thèse.

While these forms may be included in the document page count, their removal does not represent any loss of content from the thesis.

Bien que ces formulaires aient inclus dans la pagination, il n'y aura aucun contenu manquant.

Canada

ABSTRACT

A CAD-BASED MODEL FOR SITE LAYOUT

Farnaz Sadeghpour, Ph.D.

Concordia University, 2004

Front-end planning of construction sites can have a significant impact on the safety and efficiency of site operations and/or the cash flow associated with resource management. A well-planned site can: i) minimize travel time; ii) decrease time and effort spent on material handling; iii) increase productivity; and iv) improve safety, and hence decrease construction cost and time. Despite its importance, it is often ignored in the planning phase of construction projects. In practice, space allocation on construction sites is typically carried out on a first-come-first-serve basis, which could result in chaotic sites and productivity losses.

This thesis presents an interactive CAD-based model, designed to support site layout planning and to provide a framework that meets the versatility needed in actual construction practice. The developed model performs its task at two levels: site representation, and site space analysis and allocation. The site representation is carried out using object-based concepts in an open architecture. The model offers three tiers of objects: i) site objects, ii) construction objects, and iii) constraint objects. This classification assists in formalizing the representation needed for modeling construction sites. A formal structure for each tier of objects is proposed to facilitate the creation of new objects and reuse of domain knowledge. The space analysis and allocation is performed through a geometric approach, inspired by human reasoning, in search for the optimum or near-optimum location for temporary facilities.

The model has a number of interesting features: i) it provides a flexible support of a wide range of objects for site planner. This permits the set up of different construction projects via selection of objects stored in their respective libraries; ii) the model has a built-in feedback to support the development of new objects and/or updating existing ones. This allows for the gradual expansion and enrichment of the supporting libraries; iii) the geometric space analysis is visualized graphically and provides a range of ranked near optimum solutions. This feature facilitates user's comprehension of, and interaction in the layout process; and iv) the system allows site planners to define search criteria based on their knowledge and expertise. It also allows for generation of a number of what-if scenarios, utilizing different rules and comparing the final layout results.

A prototype of the developed model called CASL is implemented using Visual Basic for Applications (VBA) in AutoCAD environment. AutoCAD is dynamically linked to the relational database of Microsoft Access, which hosts the object libraries. To validate and illustrate the functionality and usability of the developed model, two case studies from the literature were analyzed using CASL. Further, a case study from industry was developed to illustrate the flexibility and other features of CASL that are not available in other models. Comparisons between the results generated by the developed model and those provided by others indicate that CASL is capable of generating layouts that better satisfy the constraints defined by planners. As such, CASL provides users with a design support tool, capable of utilizing their knowledge in generating an efficient site layout, which can aid to avoid cost overruns, schedule delays, and unsafe working conditions.

ACKNOWLEDGEMENTS

First and foremost, I wish to express my deepest gratitude to my supervisors Dr. Osama Moselhi and Dr. Sabah Alkass for their trust in me, their expert guidance, unparalleled wisdom, and invaluable support throughout my studies at Concordia. I appreciate the constructive comments and advice of the examining committee in my proposal and their time and effort for perusing and evaluating this dissertation. I would also like to extend my appreciation to Dr. Hugues Rivard for his valuable counsel and for serving in my advisory committee through the course of my studies.

I would like to thank my parents, Mahin and Karim, for encouraging me to continue my education, and for their unconditional support and understanding. I appreciate their courage in bearing the distance that separated us during my studies. Their sustained love and support was the source of strengths and inspiration, without which I could not have made it. I would also like to thank my “little” brother Farbod for being there and for cheering up my life.

I would like to extend my gratitude to Mr. John Marcovecchio of MAGIL Construction and Mr. Richard Pigatto of Drago and other professionals from the industry who have contributed to this research and have shared their valuable time and experience. I would especially like to acknowledge Mr. Germain Cardinal, the site engineer of the LG-2 project, who generously devoted considerable amount of time in developing the case

study and helped me in better understanding of the industry practice. I am also thankful to my friend Hratch Soghomonian for his extensive aid in programming and for being there whenever needed.

I would like to express gratitude to my “uncle” Dr. Hormoz Poorooshasb and his wife Pari Poorooshasb for the heartfelt care and support they offered me from my first day of arrival in Montreal. They shared my happiness and sadness during this portion of my life, for which I will always be grateful. I am also grateful to my “aunt” Zari Kousha and her husband Dr. Karim Kousha for making this place a home away from home for me.

And last, but definitely not least, I would like to thank my friends at Concordia. These years of study gave me the opportunity to attend the international institution of their friendship. I have learnt many lessons from each of them that I could not have received in any other institute. The reflection of each one can be easily tracked in my heart, my thoughts, and my life.

TABLE OF CONTENTS

LIST OF FIGURES	XI
LIST OF TABLES	XIV
ABBREVIATIONS	XV
NOMENCLATURE.....	XV

CHAPTER 1: INTRODUCTION1

1.1	CONSTRUCTION SITE LAYOUT.....	1
1.2	CURRENT PRACTICE.....	2
1.3	SCOPE AND OBJECTIVES	4
1.4	THESIS ORGANIZATION	5

CHAPTER 2: LITERATURE REVIEW 6

2.1	GENERAL CONCEPTS IN SITE LAYOUT MODELING	6
2.2	SITE LAYOUT MODELS	10
2.2.1	OPERATIONS RESEARCH.....	12
2.2.2	GENETIC ALGORITHMS.....	13
2.2.3	NEURAL NETWORKS.....	21
2.2.4	KNOWLEDGE BASED SYSTEMS.....	23
2.3	MODELS IN RELATED DOMAINS.....	31
2.4	SUMMARY.....	35

CHAPTER3: PROBLEM REPRESENTATION FOR SITE LAYOUT	
MODELING	38
3.1 INTRODUCTION	38
3.2 DATA STRUCTURE.....	41
3.2.1 SITE OBJECTS.....	42
3.2.2 CONSTRUCTION OBJECTS	44
3.2.3 CONSTRAINT OBJECTS.....	46
3.3 RELATIONSHIPS AMONG OBJECTS.....	49
3.4 OPEN ARCHITECTURE FOR SITE LAYOUT MODELING	50
3.5 GEOMETRICAL REASONING FOR SITE SPACE ANALYSIS.....	51
3.5.1 GEOMETRIC REPRESENTATION OF CONSTRAINTS	54
3.5.1.1 Closeness Relations.....	56
3.5.1.2 Distance Relations.....	58
3.5.1.3 Adjacency Relations.....	59
3.5.1.4 Orientation Relations	61
3.5.1.5 Containment Relations	62
3.5.1.6 Visibility Relation	63
3.5.1.7 Compound Relations.....	67
3.5.2 SITE SPACE ANALYSIS CONSIDERING MULTIPLE CONSTRAINTS	69
3.6 SUMMARY.....	71
 CHAPTER 4: PROPOSED MODEL	 74
4.1 INTRODUCTION	74
4.2 MODEL ARCHITECTURE.....	76
4.3 DATABASE	78
4.4 PROJECT MODULE.....	82

4.4.1	SITE MODULE	83
4.4.2	CONSTRUCTION MODULE	85
4.4.3	CONSTRAINT MODULE	87
4.4.4	QUEUEING MODULE	89
4.5	LAYOUT CONTROL MODULE	96
4.5.1	SPATIAL ANALYSIS MODULE	97
4.5.2	LOCATING MODULE	100
4.5.3	EVALUATING MODULE	101
4.6	SUMMARY.....	104

CHAPTER 5: IMPLEMENTATION 106

5.1	IMPLEMENTATION ENVIRONMENT: AUTOCAD-VBA-ACCESS INTEGRATION.....	106
5.2	IMPLEMENTATION ISSUES	109
5.2.1	AUTOCAD OBJECT MODEL.....	109
5.2.2	EMPLOYED GRAPHICAL OBJECTS.....	111
5.2.3	OBJECT IMPLEMENTATION IN CASL	116
5.2.4	DEVELOPED FUNCTIONS IN CASL	118
5.3	IMPLEMENTED MENUS.....	124
5.3.1	PROJECT SETUP MENU	126
5.3.2	LAYOUT MENU	137
5.4	SUMMARY.....	143

CHAPTER 6: CASE STUDIES AND PERFORMANCE EVALUATION.. 145

6.1	INTRODUCTION	145
6.2	CASE EXAMPLE 1: SINGLE FACILITY, SINGLE CONSTRAINT.....	145

6.3	CASE EXAMPLE 2: MULTIPLE FACILITIES, SINGLE CONSTRAINT	151
6.4	CASE EXAMPLE 3: MULTIPLE FACILITIES, MULTIPLE CONSTRIANTS.....	159
6.4	SUMMARY.....	175
CHAPTER 7: SUMMARY AND CONCLUDING REMARKS.....		177
7.1	SUMMARY.....	177
7.2	CONTRIBUTIONS.....	182
7.3	LIMITATIONS OF THE MODEL.....	183
7.4	FUTURE WORK.....	184
REFERENCES		187
APPENDIX I		200
APPENDIX II.....		206
APPENDIX III.....		211

LIST OF FIGURES

Figure 2.1.	Organization of site layout models	6
Figure 2.2.	Site space representation in the model developed by Li and Love (1998)	15
Figure 2.3.	Site space representation in SitePlan (Yeh 1995)	22
Figure 3.1.	The structure of constraint object	48
Figure 3.2.	Conceptual relations among site layout entities in UML notation	50
Figure 3.3.	Staircase Function (a) Ascending (b) Descending	55
Figure 3.4.	Step function (a) Falling step (b) Rising step	55
Figure 3.5.	Graphical representation for closeness relation (a) Polar closeness (b) Linear closeness	57
Figure 3.6.	Graphical representation for distance relation	59
Figure 3.7.	Graphical representation for adjacency relation	60
Figure 3.8.	Graphical representation for orientation relation (a) East/West relations (b) North/South relations	61
Figure 3.9.	Graphical representation for containment relations	63
Figure 3.10.	Visibility situations based on the relative height of objects	65
Figure 3.11.	Graphical representation for visibility relation (a) and (c) if $h_M > h_P > h_N$; (b) and (d) if $h_N > h_P > h_M$	66
Figure 3.12.	Compound relation (a) Descending step function for distance relation (b) Descending stair function for closeness relation (c) Compound function (d) Graphical representation for the compound relation	68
Figure 3.13.	Rings representing the closeness to three objects divide the site area into smaller segments	70
Figure 3.14.	Dividing site area into smaller segments (a) Rings representing three constraints, (b) Site segments before intersection, (c) Site segments after intersection	71
Figure 4.1.	Model Architecture	77
Figure 4.2.	Connectivity between model's database and CAD database through object ID	78
Figure 4.3.	Defining objects (a) Selecting from library (b) Creating new objects	80
Figure 4.4.	Defining constraining objects	81
Figure 4.5.	Entity relationship in the database	82
Figure 4.6.	Order of activities in Project Module	83
Figure 4.7.	Generating geometry of objects	84
Figure 4.8.	Dataflow in site and construction sub-modules	86
Figure 4.9.	Dataflow in the constraint sub-module	88

Figure 4.10.	Independent queuing of objects in the Queuing Module.....	91
Figure 4.11.	Alignment of objects (a) Corner-to-corner adjacencies (b) Vertical and horizontal alignments.....	100
Figure 4.12.	Dataflow in the developed model.....	103
Figure 5.1.	AutoCAD ActiveX technology (Autodesk 2001).....	108
Figure 5.2.	Partial object model of AutoCAD (Autodesk 2001)	110
Figure 5.3.	Relationship diagram of the database	118
Figure 5.4.	<i>Rings</i> function (a) Offsetting central circle, copy offset circles, send them to array of circles, convert to array of region objects (b) Subtracting consecutive circles (c) Subtracting the rings from site boundary	120
Figure 5.5.	<i>InterSub</i> function	121
Figure 5.6.	Updating <i>nonOverlap</i> pairs	124
Figure 5.7.	“Project Setup Menu” form	126
Figure 5.8.	“Select Drawing Style” form.....	127
Figure 5.9.	“Insert” form from AutoCAD.....	127
Figure 5.10.	“Aided Drawing” form	128
Figure 5.11.	“Define Construction Objects” form	130
Figure 5.12.	“Object Properties” form	131
Figure 5.13.	“Modify Construction Object” form.....	132
Figure 5.14.	“Write Block” form from AutoCAD	133
Figure 5.15.	“Select Constraint Objects” form	134
Figure 5.16.	“Assign Constraints” form.....	135
Figure 5.17.	“Queuing Construction Objects” form	136
Figure 5.18.	“Layout Menu” form	138
Figure 5.19.	“Calculate Utility Score” form	140
Figure 5.20.	“Locating Construction Object” form	141
Figure 5.21.	“Refinement” form	142
Figure 6.1.	Layout of the manufacturing office and analyzing grid	147
Figure 6.2.	Total weighted distance graph for the corridor area	150
Figure 6.3.	Geometric analysis of corridor area for water fountain problem.....	151
Figure 6.4.	Permanent facilities and existing site objects (Mawdesley et al. 2002)	152
Figure 6.5.	Available and unavailable site area	153
Figure 6.6.	Site analysis for temporary office.....	154

Figure 6.7.	(a) Layout 1, suggested by Mawdesley et al. (2002); (b) Layout 2, generated considering the weights of permanent facilities; (c) Layout 3, generated considering weights of temporary and permanent facilities; (d) Layout 4, generated considering weights of temporary and permanent facilities	156
Figure 6.8.	Actual site layout of LG-2 project.....	160
Figure 6.9.	Site analysis for office	164
Figure 6.10.	Site analysis for garage.....	165
Figure 6.11.	Site analysis for formwork storage area	166
Figure 6.12.	Site analysis for welding shop	167
Figure 6.13.	Site analysis for plumbing workshop	167
Figure 6.14.	Site analysis for storage.....	168
Figure 6.15.	Site analysis for electrical shop	169
Figure 6.16.	Site analysis for equipment parking	170
Figure 6.17.	Site analysis for soil and concrete test labs	170
Figure 6.18.	Site analysis for parking	171
Figure 6.19.	Site analysis for steel storage.....	172
Figure 6.20.	Site analysis for electrical storage	172
Figure 6.21.	Site analysis for lab parking	173
Figure 6.22.	Site arrangement for LG-2 generated using the proposed	175
Figure III.1	View of LG-2 project site.....	211
Figure III.2	Spillway.....	211
Figure III.3	South-east view of the site.....	212
Figure III.4	South view of the site	212
Figure III.5	View of garage.....	213
Figure III.6	View of storage, garage, and equipment parking	213

LIST OF TABLES

Table 3.1.	Properties of site objects.....	43
Table 3.2.	Properties of construction objects.....	45
Table 3.3.	Properties of constraint objects.....	49
Table 3.4.	Classification of the considered relationships	53
Table 4.1.	Composing elements for the constraint objects.....	93
Table 4.2.	Queuing construction objects using Max LCE rule.....	94
Table 4.3.	Queuing construction objects using Min UCE rule.....	94
Table 4.4.	Conflict on Max LCE. Case 1- Select A.....	95
Table 4.5.	Conflict on Max LCE. Case 2- Select F	95
Table 5.1.	Polyline object properties	112
Table 5.2.	Polyline and region object methods.....	113
Table 5.3.	Region object properties.....	115
Table 5.4.	Typographical conventions.....	125
Table 6.1.	Involved departments and related data (Zouein 1996)	146
Table 6.2.	Comparison of results using Euclidean and rectilinear measurements.....	149
Table 6.3.	Closeness weights for four construction objects, adapted from Mawdesley et al. (2002)	153
Table 6.4.	Total utility score breakdown for layouts in Figure 6.7	158
Table 6.5.	Data pertaining queuing of construction objects for LG-2 project.....	160
Table 6.6.	Constraint objects defined for the LG-2 project	162

ABBREVIATIONS

2D	Two-dimensional
3D	Three-dimensional
4D	Four-dimensional
AI	Artificial Intelligence
CAD	Computer Aided Design
GA	Genetic Algorithm
GIS	Geographic Information Systems
GUI	Graphical User Interface
KBS	Knowledge-Based System
NN	Neural Networks
OR	Operations Research
SQL	Structured Query Language
UML	Unified Modeling Language
VB	Visual Basic
VBA	Visual Basic for Applications

NOMENCLATURE

LCE	Located Constraining Elements
ML	Model Library
PP	Project Palette
Q	Multi-Attributed Queuing Score
U	Utility Score
U_t	Total utility score
UCE	Unlocated Constraining Elements

CHAPTER 1

INTRODUCTION

1.1 CONSTRUCTION SITE LAYOUT

Space on construction sites is recognized as a resource that is as important as other resources of money, time, material, labor, and equipment (Tommelein et. al 1992c, Hegazy and Elbetagi 1999). Despite its importance, space was not typically modeled in construction management tools a decade ago (Zouein and Tommelein 1993), and the situation has not changed since. Studies have shown that for every dollar spent on pre-planning of large projects, four dollars could be saved in their total cost (Hegazy and Elbetagi 1999). Proper site planning can improve the efficiency of construction operations and reduce material handling costs, which consumes up to 40% of site expenses (Zouein 1996). Thus, detailed studies of site arrangements before commencement of construction can have an impact on the efficiency of site operations and the cash flow associated with resource management.

However, site visits and interviews with superintendents and site managers, as well as literature (Elbetagi and Hegazy 2003, Tommelein and Zouein 1993a), reveal that site layout planning is often ignored in the planning phase of construction projects. A good portion of productivity losses occur due to space-related issues such as unnecessary travels on site for tools and materials, searching for misplaced tools or materials, and work interruption due to space congestion (Cheng 1992). A well-planned site can

contribute in decreasing productivity losses by minimizing travel time, decreasing time and effort spent on material handling, and in improving safety. As well, planning for site layout aids in detection of potential space conflicts, which in turn can increase certainty to schedules and reduce the risk of project cost and time overrun (Dawood et al. 2003). Further, the algorithm proposed for the construction site layout optimization can be adapted for other applications such as space scheduling and manufacturing cell-layout (Harmanani et al. 2000).

1.2 CURRENT PRACTICE

Along the literature review, the current practice of site layout was studied through site visits and interviews with practitioners, specifically of construction projects in the Quebec region. Despite the acknowledgment of practitioners of the impacts of site layout on efficiency of site activities, it is often ignored in the planning phase of construction projects. This means usually no drawings are prepared for site layout purpose. At best, site managers or superintendents draw sketches on site plans generated for other purposes (e.g. landscape plans). As construction progresses, information is added to the first drawing, so the drawings consist of several layers of information. As these layers of information are added along the progress of construction, they become more complicated and difficult to understand. In larger projects, templates, two dimensional scale models, and in some cases three-dimensional physical models are used to represent the layout of the site. These representations aid in visualization of the site arrangement and sizing of facilities, but they do not provide space allocation methodology or adequate information about the site. In practice, space allocation on construction sites is typically carried out on

a first-come-first-serve basis and mainly through human judgment, which could result in chaotic sites and may give rise to productivity losses and safety related incidents.

This reluctance towards site planning can be attributed to a number of reasons. For one, site layout is not considered as a defined task in the planning, design and construction phases of a project. Most owners are reluctant to spend money on site layout due to their lack of awareness of its importance and its effect on construction performance and productivity. In the design phase, allocating time to the site layout task increases engineering man-hours, and hence the cost, thus it is preferably left for the field practitioners to plan it. In the construction phase, site planning expenses are usually charged to project overhead which has to be kept low in order to remain competitive (Cheng 1992).

Another reason can be associated to the ill-structured nature of site planning. To prepare a layout for a site, the planner has to extract information from various sources such as drawings, spreadsheets, bar charts and text. This makes the process of data collection tedious and time consuming. Also there is no standard method or conventional tool in the market to assist site planners in the task (Riley and Sanvido 1997). Site layout planning is mainly carried out based on human judgment; to the extent that some even consider it an art rather than a science (Tam and Tong 2003). On the other hand, the uniqueness of each project and its conditions result in a great variation in site layout strategies. As such, there is a strong need for an approach to allow site planners to apply their individual knowledge due to different training and background, while aid in reducing human errors.

1.3 SCOPE AND OBJECTIVES

The main objective of this research is to develop a framework and a methodology for efficient planning of construction site layouts.

The sub-objectives of the research are:

- To identify and classify the entities required to develop a site layout model.
- To formalize the representation of such entities for computer representation.
- To develop an effective space analysis and object-locating optimization strategy suitable for planning construction sites, and compliant with practitioners' approach in this domain.
- To implement the proposed methodology in an interactive computer-based site layout model.

The scope of this research focuses on two-dimensional analysis of sites, and as such, the topographic features of a site are not considered. As well, site layout is modeled as a static problem. Hence, modeling the changes that occur on construction sites through the course of time is out of the scope of this research.

1.4 THESIS ORGANIZATION

This thesis consists of seven chapters. Chapter two provides an overview of existing site layout models. It focuses on problem representation and problem solving approaches utilized in these models, highlighting their capabilities and limitations, and summarizing the gaps in the literature. The proposed formalized structure for site layout problem representation and the developed methodology for geometric analysis of site space and its visualization are described in chapter three. Chapter four presents an interactive CAD-based site layout model, describing its architecture and the interconnectivity among its modules. Chapter five presents a prototype of the developed model in an integrated AutoCAD-VBA-Access environment. Chapter six presents three case examples analyzed using the developed model to illustrate its functionality and accuracy. Chapter seven includes a summary of the work along with its contributions, limitations, and recommendations for future work.

CHAPTER 2

LITERATURE REVIEW

2.1 GENERAL CONCEPTS IN SITE LAYOUT MODELING

The development of tools to aid planners in the task of site layout has been a topic of research in this area. Due to its complexity, from the early studies, a computerized approach deemed essential to implement the process of site layout. Various site layout models have been developed using different approaches and assumptions. This chapter provides an overview of a number of site layout models and the concepts utilized in developing them. To facilitate the study and comparison of these models and concepts an organization scheme, adopted from Tommelein et al. (1992b), is utilized in this chapter and illustrated in Figure 2.1.

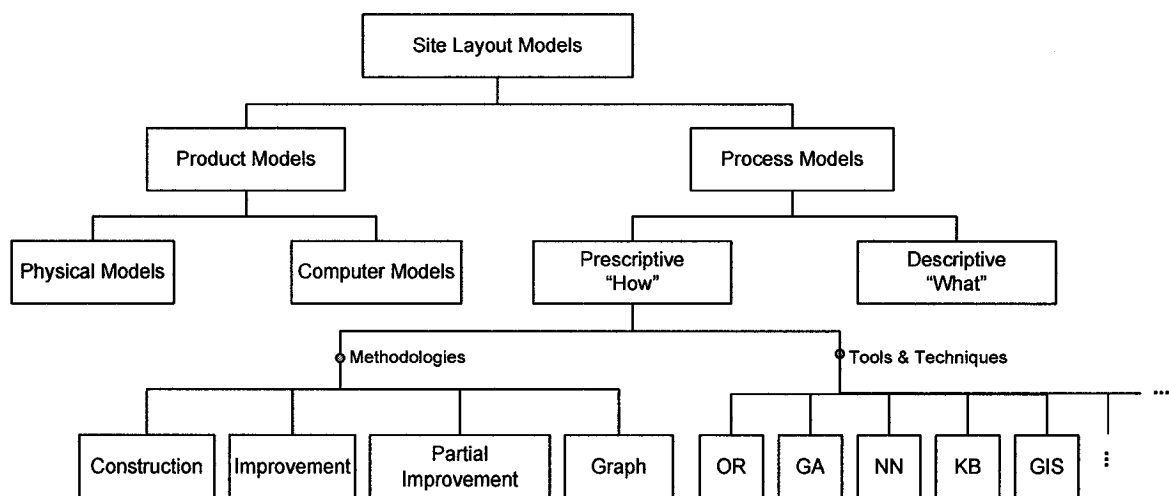


Figure 2.1. Organization of site layout models

Based on the task they perform, site layout models are classified in two categories: product models and process models. Product models define the information structure of the constituent parts of a subject in terms of form, behavior and the relation between parts and assemblies (Eastman 1999). The purpose of previously developed site layout product models was mainly limited to the visualization task (Tommelein et al. 1992b) and no formal structure for the site layout representation were introduced. These visual or physical models assist the site planner in visualizing the facilities and their location on site. Since real objects cannot be actually manipulated on site to search for the best layout, iconic models are used to perform this task. The most commonly used product models used in site layout practice are engineering drawings, templates, previously designed sites, physical scale models and three-dimensional (3D) CAD models (e.g. Thabet and Beliveau 1993, Thabet et al. 1992).

When using drawings for site layout planning, sketches of facilities on paper are overlaid on the site map and moved around until the planner obtains a satisfactory layout. Engineering drawings are the most commonly used product models for preparing site layout designs; however, they are difficult to update. Templates are also used widely by site planners in practice. Templates are cardboard cutouts representing the facilities that are moved around on the site drawing until a satisfactory layout is obtained. This approach was later modeled in a computer system by Rad (1982). The system allows planners to generate site perimeters and select a set of facilities from a list. Selected facilities are graphically displayed on the screen and the system identifies potential

overlaps among facilities. The planners can then resize or relocate facilities until a satisfactory layout is reached.

Physical 3D models are more expensive methods for providing a spatial representation for facilities on site space. High cost and difficulty in maintenance and updating are the main reasons that have prevented physical 3D models from being commonly used in practice. However, recent advances in information technology and accessibility of computers have caused the rapid widespread use of 3D computer aided models. Unlike physical models, these models do not require maintenance and are easy to update. More importantly, it is possible to reuse and adapt these presentations for other projects. In summary, the existing product models help site planners to visualize the problem. While product models are helpful in exploring different possibilities of a layout, they do not provide a methodology for locating the facilities, nor do they give an evaluation scale to rank different layout configurations.

Process models address the site layout problem with regards to *performance*. Process models can be descriptive or prescriptive. Descriptive process models describe “what” should be done, but not “how” it can be done (e.g. Handa and Lang 1988 and 1989, Rad and James 1983, Halpin and Woodhead 1998, Toole 2002). They provide a list of what should be considered during the layout process, but they are limited when it comes to providing a procedure on how the process should be performed and in what order. Prescriptive process models address the “how” question by providing a methodology for the process at different levels. For example, Rad (1980) has developed a method to

measure the congestion in different areas on site. The site is divided into working areas by user and for each site segment, congestion is calculated based on the space requirements for the resources assigned to the activities that take place in that segment. To provide a base for comparison, the space requirements for equipment, machinery, and labor is expressed in equal number of men (EM) multiplied by the duration of the activity. Accordingly, the congestion for different site areas can be compared. If the space requirements for a site area are greater than the site itself, then the schedule should be modified. While this system prescribes a methodology to measure the congestion on site, it does not provide a method for positioning facilities on site. There are several other prescriptive process models that describe the locating procedure and layout strategies. A number of these models are discussed in the next section.

Two methodologies have been used in large to develop prescriptive layout models: improvement methods and construction methods (Moore 1980, Li and Love 1998, Sirinaovakul and Thajchayapong 1996). Improvement methods start from a complete initial site design, which is then altered by moving objects around the site to create new layouts. Each layout is then evaluated against an objective function and accordingly the best solution is identified (e.g. Elbeltagi and Hegazy 1999, Moore 1971). Construction methods on the other hand, locate the objects one at a time. In each step, one object is located on the site, considering the updated status of the site before adding that object. A median placement method is partial improvement, which has the characteristics of both of the above-mentioned algorithms (Li and Love 1998). Here, a facility is located at all of the possible positions and a set of possible partial layouts are generated and evaluated. In

this way, the most suitable alternative location for the facility at hand is identified. The process is then repeated until all facilities are located. A fourth methodology employs the graph theoretic techniques to decide on adjacency between pairs of objects (e.g. Foulds 1983, Foulds et al. 1985). This method is mainly used for floor planning in architecture. The main problems with this method are that it does not guarantee to reach a layout in the end, and that only adjacency relationships can be expressed (Tommelein et al. 1992b).

2.2 SITE LAYOUT MODELS

Developing layout models originally started by utilizing mathematical optimizations and gradually moved towards heuristic models. The main reason for this tendency, apart from the excitement of exercising heuristics, was that only a few of these mathematical models seemed to be successful and even then, the models worked only for the specific case introduced in the problem (Tommelein 1992).

A number of optimization procedures were introduced in earlier studies. However, these approaches were rendered computationally unfeasible for large problems due to the limitations of computer technology at the time (Hamiani 1987). They also required great skill from the user to fit a given problem into the model. In these optimization methods, site facilities were generally dealt with as points, and thus their dimensions were ignored. This is referred to as *location problems* in the literature, as opposed to *layout problems* (Tommelein et al. 1992a). In location problems, the space limit is not an issue, so it is appropriate to represent objects as points. As such, location methods are more suitable for locating a single facility like cranes or batch plants. In spite of several research efforts in

developing quantitative techniques for site layout in the past, their use was rather limited. The employed techniques were difficult to learn and use, and their computer implementations were complex, and hence received resistance towards being commonly used (Hamiani and Popescu 1988). As well, at the modeling stage these techniques normally used a degree of simplification from real practice, resulting in loss of information (Tam and Tong 2003). Later, interest leaned towards heuristic procedures. The heuristics for site layout problems consist of knowledge prescribing the order of facility selection and meeting the constraints while locating them on site.

Artificial intelligence (AI) systems have received considerable interest in recent decades and have been used in various research areas including space planning and site layout (Smith et al. 1995). In particular, Knowledge-based Systems (e.g. Hamiani (1987), Tommelein et al. (1991 and 1992b)), Neural Networks (e.g. Yeh 1995), and Genetic Algorithms (e.g. Philip et al. (1997), Li and Love (1998), Hegazy and Elbeltagi (2000), Harmanani et al. (2000)), have been used to develop site layout models. As well, other hybrid models have been developed to exploit the benefits from features offered by different techniques. For example, MoveSchedule (Zouein 1996) is a dynamic space scheduling system that incorporates expert's knowledge in its optimization process. ArcSite (Cheng 1992) integrates a knowledge-based approach in a model developed using GIS. Another model developed by Elbeltagi et al (2001) uses Knowledge-based Systems (KBS), fuzzy logic, and Genetic Algorithms (GA) to generate schedule-driven site layouts. The main notion in AI systems is their ability to deal with inexact, missing, or poorly defined problems. Compared to numerical methods, AI is designed to deal with

qualitative features, which are of high interest for site layout problem. However, AI systems do not provide optimal solutions (Luger and Stubblefield 1989).

In this section, a number of existing models are described, and the advantages and disadvantages of each are discussed. This discussion focuses on modeling issues including problem structure and representation, the procedure of locating facilities on site, and the constraints considered in locating facilities on a construction site. For convenience in the discussion and comparison, these models are classified into four groups according to the dominant techniques used in analyzing the site space. Obviously, this is not a concrete classification and some models fall in the overlap areas between the four groups since they are developed as an integral part of an IT-based system that consists of other tools.

2.2.1 Operations Research

Prior to the construction industry, site layout was introduced in industrial engineering to find optimum layout for departments on a manufacturing site, mainly using Operation Research (OR) techniques (Scribin and Vergin 1975). OR models represent the behavior of systems by means of numerical equations and constraints (Tommelein et al.1992a). CORELAP (Lee and Moore 1967, Moore 1971) was one of the earliest attempts to develop a site layout system. The system was designed to layout a set of predetermined departments on an industrial site and calculates a score for the alternative layouts against an objective function. In this model the objective function reflects the distance between pairs of departments and their respective closeness ratings. The system is designed to

work interactively with the planner and takes the flow of material, activity relationship (closeness weight) and space requirements for each department as input. In locating facilities, CORELAP ranks them in descending order of closeness relationship. The system then suggests a layout by applying OR techniques to find a location for each facility and calculating the respective score for the layout. The user can later use this layout as an improvement template and shift the icons to the desired location on site while the system updates the design score according to the new layout.

The main disadvantage of the OR site layout systems is that they demanded a large amount of data to run (Tommelein et al. 1992a) and their locating constraints were limited to minimizing distance. This makes such models useful only so far as distance is concerned whereas in real practice, the quality of a proposed layout is judged by a multitude of criteria.

2.2.2 Genetic Algorithms

The application of genetic algorithms (GAs) to construction site layout is less than a decade old. The basic notion in GA techniques is to mimic the Darwinian principal of “survival of the fittest” to solve complex optimization problems. Unlike other optimization algorithms that work with one solution at a time, GAs work with a family of solutions (Harmanani et al. 2000). The first population is generally initialized randomly or designed roughly by the user. This is called the “initial population”, from which the “next generation” of solution is derived. Each time a new offspring is generated, the fitness of each individual is evaluated against the objective function. A new individual is

supposed to be a progressively better solution as the new generations are generated. Hence, GA-based layout models utilize the improvement method. In implementing site layout problems using GAs, the facilities and their location on the job site are represented as genes on a chromosome string. In other words, each solution is designed as a vector whose length is equal to the number of facilities being considered in the design. Several GA operations such as crossover, mutation, and reproduction are applied to generate the offspring.

The first attempt to utilize GAs for construction site layout was conducted by Philip et al. (1997). In this research the site space is scaled to a square grid and facilities are represented as rectangles. The top-left coordinates of the facility address its location in the string representation of the GA. The system introduces the buildings/plant and their relative components as permanent facilities (PF), while temporary facilities (TF) are those which have been put up only during the construction phase, and hence subject to location. To overcome the problem of overlapping, a penalty function is added to reduce the fitness of any individual layout that contains overlapped areas.

This study identifies eight typical factors affecting a layout are identified as travel frequency between facilities, access to roads/utilities, nature of usage, space requirements, type of terrain, duration of facility requirement, construction schedule, and human factors. Despite this recognition, the only constraint implemented in the system is minimizing the travel effort between a pair of facilities on site. This can be partly attributed to the string format of GAs that makes representation of other spatial objectives

difficult. Similarly in a model developed by Li and Love (1998), the objective is to minimize the travel effort between facilities on site. This effort is calculated on a daily basis by multiplying the distance between facilities and the frequencies of trips in one day. Based on this objective, the model attempts to locate a set if predetermined facilities in a set of predetermined locations on site (Figure 2.1).

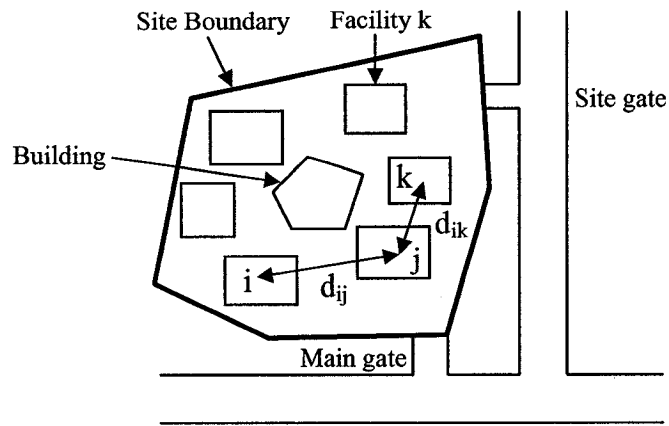


Figure 2.2. Site space representation in the model developed by Li and Love (1998)

The conceptual representation scheme of the system is a permutation matrix, in which each layout is represented by an $n \times n$ matrix (n being the number of facilities). Since for larger projects with numerous facilities, the matrix occupies a considerable space. Hence, the permutation matrix is implemented in string format. In this representation, the position of an element on the string represents the facility number, while its value represents the facility location. A useful modeling feature in this research is the ability to represent fixed facilities. This feature is achieved by assigning and fixing a gene in the “chromosome” to a fixed facility (e.g. gates). During the process of generating new offspring, the genes containing the fixed facilities are omitted from participating in the process. When a new string is generated, fixed facilities are added to it since their relative

location affects the fitness of the entire layout. However, the location of the building, which has a large effect on the overall fitness score of the layout, and hence the relationships, has not been considered in this study. The process of searching for the optimum layout will end when the fitness difference between the best and worst members of the population is smaller than an allowable value.

Considering a set of predetermined locations on site is very limiting compared to construction practice in reality where site planners are able to locate construction facilities anywhere on the site. The predetermined areas on the site are defined large enough to accommodate the largest facility. The problem with this representation arises in locating smaller facilities where a large space is reserved, and hence wasted. The predetermined locations are represented in rectangles, although, the site boundary can take other shapes. This can count as one step towards flexibility in shapes (Fig. 2.1).

EvoSite (Elbeltagi and Hegazy 1999 and 2003, Hegazy and Elbeltagi 1999 and 2000) is another GA-based site layout model that exhibits more flexibility in representing the shape of facilities and site boundary. EvoSite is a simplified site layout model implemented on the grid structure of a spreadsheet. The site boundary is generated following the grid of the spreadsheet, and hence non-rectangular shapes can be defined. To better represent oblique lines, the model was later adapted to allow the use of different orientations for the grid, selected based on the orientation of the majority of facilities (Elbeltagi and Hegazy 2003). This feature renders EvoSite more flexible than the previous models. However, when representing curved lines, the grid is not a fine solution.

The size of the grid is selected as a function of the facilities' size. In an earlier version of EvoSite (Hegazy and Elbeltagi 1999) for example, the size of the site grid was calculated according to the greatest common divisor (g.c.d.) of all facility sizes. In this way, each facility is represented as a number of site grids. Later in this research, the grid size was changed to be equal to the area of the largest facility with each facility occupying one grid (Hegazy and Elbeltagi 2000). As in the previous model, the problem with this approach is in locating smaller facilities, where a larger site area is considered occupied when only a small part of the grid is actually in use.

Another level of improvement in EvoSite, in comparison to the previously described models, is the definition of three categories for objects on site: facilities, which are to be located in available cells; fixed facilities that have user defined fixed locations on site but still carry the relationship weight with other facilities; and obstacles that represent non-allocable areas on site. A numerical proximity weight is used to represent the closeness relationship, referred to as a qualitative method. The closeness relationship is calculated based on a quantitative value such as transportation cost per unit of time or the amount of transferred material on site. Six closeness relationships between pairs of facilities (absolutely necessary, especially important, important, ordinary closeness, unimportant and undesirable) are taken from the literature and set prior to the optimization (Hegazy and Elbeltagi 2000). The user can assign desired values to each category, as well as the number of offspring generations, after which the improvement process stops. Once the layout is designed, it can be used as a template to study the effect of relocating facilities

on the objective function. Each time a facility is relocated, the total travel distance is calculated.

A subsequent study added dynamic capabilities to EvoSite by considering construction schedules in creating site layouts (Elbetagy et al. 2001). The model creates several layouts for user-defined time intervals. At each time interval, the model calculates new space requirements and generates a layout. In generating new layouts, the model considers reusing the space previously occupied by facilities that are no longer needed on site. As well, it utilizes parts of constructed space to accommodate temporary facilities. Nevertheless, similar to previous GA-based layout models, the objective of the model is to minimize total travel distance.

In a more recent study, Mawdesley et al. (2002) have developed a model that encodes other constraints in its objective function. In this model, the fitness of the GA is formulated as a summation of material transportation cost, facilities setup cost, facilities removal cost, and personnel visit cost. The model developed by Mawdesley et al. (2002) is similar to EvoSite in dividing the site area into grids. For each site grid, a cost is calculated based on setup, removal, and travel costs. Some layout constraints are implied by means of these three cost types. For example, in order to deter the consideration of an unavailable site area, an arbitrarily high setup cost is assigned to the corresponding grid cells. Based on the cost distribution on the grid cells, a least cost route between facilities is used as a criterion for facility layout. The shape of facilities in this model follows the grid: permanent facilities are represented by a number of grid cells, and temporary

facilities are assumed to occupy one grid. Although using a grid seems to be compliant with the implementation of GAs, in this model, as well as in EvoSite, it imposes rigidity on the shape of the facilities.

A model presented in (Harmanani et al. 2000, and Zouein et al. 2002) defines GA-based site layout without using a grid system. As such, the location of facilities, or their size, is not constrained to the grid. Yet, facilities are defined only in rectangular forms. A closeness weight is assigned to pairs of facilities, measured based on flow or unit transportation cost between them. An interesting feature of this model is its ability to imply a minimum or maximum distance between facilities and constrain a facility to have the desired surrounding facilities on its four main sides (i.e. North, South, East, West). Also, the orientation (0 or 90 degrees) and non-overlap constraints can be applied to the facilities. The model utilizes a larger number of GA operators to achieve better results, and implements constraints other than minimum travel distance. However, the model is not applicable for site layout. The layout in this model is defined as a bin-packing-like problem. Optimally, this class of problems results in tightly packed layouts. In construction sites, facilities are not tightly packed, and even in congested sites, spaces are left between facilities to provide access. Also, the model does not introduce a method to implement fixed facilities on site in order to setup conditions of construction sites.

The model was tested on three problem cases: 1) equal-sized rectangular facilities with equal proximity weights among all facilities (weight = 1) and no constraints; 2) unequal-sized rectangular facilities with various closeness weights among facilities and no

constraints; and 3) similar to case 2, with constraints defined for the facilities. An interesting feature of this case application was to study the impact of total objects-to-site-area ratio. It was interesting to note that the quality of layouts dropped dramatically when the objects-to-site-area ratio passed 50-60%. Also the model showed better results in the first case compared to the second and third cases. The authors concluded that *GA performs better in loosely constrained problems with small objects-to-site-area ratio, even with larger number of objects* (Zouein et al. 2002).

In a hybrid GA-NN location model developed by Tam and Tong (2003), GA is used to determine the optimum location for the tower crane and supply points, in regards to a set of demand points. Neural networks (NN) are used to model the operation of the crane and predict the hoisting times, which consist of the supply and return time. As in the case of location problems, the crane, supply and demand items are represented as points. The chromosome string consists of permissible locations for the supply points, the identified locations for the demand points, and possible location of the tower crane. The objective function is defined as the minimum of total hoisting time, quantity of material flow, and cost of material flow between pairs of supply and demand points. Tam and Tong (2003) consider the setting of GA parameters, such as population size, probability of crossover and mutation, as a trial and error process. The values for these parameters rely heavily on the knowledge and experience of the user and are set based on previous work. The rate of convergence and avoidance of local optimum are also reported to be factors that require consideration. Although such GA-related factors affect the final layout, they are not easy to set and the user needs to be an expert in GA to be able to adjust such factors to the

advantage of the layout. For example, in a case study conducted by Li and Love (1998), the effect of the population size on the convergence of the developed layout system was investigated. In a study with eleven (11) facilities, among population sizes of 30, 50, 70, 100 and 150, the population with the size of 100 had the best result. This example illustrates the need for a good GA knowledge for site planners in order to work with GA-based site layout models.

2.2.3 Neural Networks

Neural networks are a family of massively parallel architecture that solve problems by the cooperation of simple, but highly interconnected computing elements called neurons (Wasserman 1989). The main drawback of traditional NN for optimization problems was getting trapped in local optimum. Simulated annealing is a probabilistic hill-climbing search algorithm that was proposed to solve combinatorial optimization problems. Random changes are allowed in the simulated annealing technique to escape the local optimum; however, it requires unacceptably long computation times (Yeh 1995). SitePlan (Yeh 1995) is a site layout model that applies a hybrid type of Neural Networks called Annealed Neural Networks (ANN). ANN inherits features of both NN and simulated annealing; like simulated annealing, it does not get trapped in local minimum, while exhibiting rapid convergence of the network. SitePlan represents site layout as a problem of finding the best location for a set of equal size rectangles in a set of predetermined locations on site (Figure 2.2). To encode such a representation, SitePlan uses an $n \times n$ permutation matrix, in which n refers to the number of facilities and predetermined locations on site. This representation is obviously an extreme simplification of the reality

of construction sites where facilities come in all shapes and sizes and can be placed anywhere on the site.

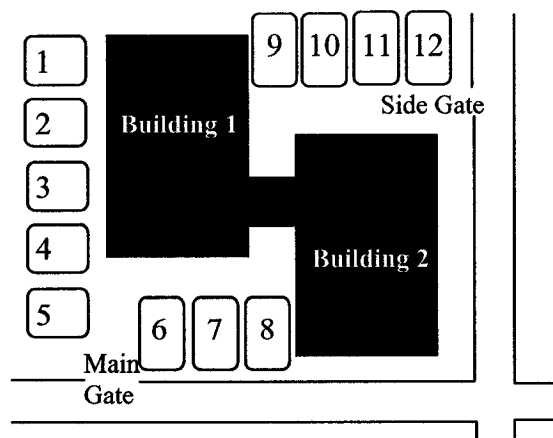


Figure 2.3. Site space representation in SitePlan (Yeh 1995)

The main advantage of NN for the site layout problem is that the structure of NN is capable of easily accommodating multiple constraints. However in SitePlan, the problem has been formulated as the sum of two costs: construction cost (i.e. cost of assigning a facility on one of the predetermined locations on site) and interactive cost (i.e. interactive cost of assigning facility X on a location neighboring facility Y). There is also a penalty factor defined to prevent two facilities from occupying the same location. These costs should be determined by the user at the setup stage. The suggested factors to be considered include adjacency of the facilities, the distance between them, availability of space, their position relative to other facilities and view from other facilities. Such predetermination of the construction and interactive cost are not easy and require professional experience. As well, just as in the case of GAs, a high level of expertise is necessary to adjust ANN-related setup factors such as initial temperature. While these

factors affect the quality of the layout and speed of convergence, they can be confusing for site planners with little or no ANN knowledge.

2.2.4 Knowledge Based Systems

The notable advances in knowledge-based reasoning in the past decades have provided the ability to focus on representation and reasoning in layout problems (Chinowsky 1991). Expert Systems are knowledge-based (KB) programs in which heuristic strategies developed by human experts are used to solve domain specific problems. Knowledge structure and representation has a direct influence on the performance of the KB system. Building the knowledge base requires a description of problem-specific, as well as relevant design knowledge. This knowledge includes theoretical information, as well as practical domain knowledge, which is extracted from human domain experts. As a result, these systems promise to provide solutions that have the same quality as domain expert solutions (Luger and Stubblefield 1989). The main advantage of KBESs is that they enable researchers to address problems for which no algorithmic solution exists. In this section, modeling details of four construction site layout knowledge-based systems will be discussed.

Consite (Hamiani 1987 and 1989, Hamiani and Popescu 1988) is one of the earliest applications of expert systems to solve construction site layout problems. The system has been developed in KEE (Knowledge Engineering Environment) using a mixture of rules, frames and object-oriented programming, and uses a plan-generate-and-test strategy to locate the temporary facilities on site. During the planning phase, the order in which

objects enter the site is decided based on priorities given to the facilities by the experts. This priority is awarded according to the frequency of trips from the work area to the facility. Once the order is decided, Consite activates the generator. The generator identifies the possible locations for facilities through space representation used in Consite. The satisfaction of constraints for a selected location is thereafter tested, and the final layout representation is updated accordingly. Consite keeps track of all location alternatives in case of dead ends.

Space is represented through convex polygons in Consite. The site is represented by means of a set of polygons with three labels: open space, access space, and occupied space. Facilities are also represented in convex polygons of two types: closed and access. The user has to select the required facilities from the facility library and specify the desired size. The ability to use different sizes for facilities is an advantageous feature in Consite, making it more flexible in facility selection. The number of polygons representing the site increases as more facilities are located on site, and hence the runtime increases as the design advances, which may lead to unacceptable runtimes. The main advantage of Consite is the ability to define multiple constraints for facilities. These constraints include checking for distance between objects, adjacency of objects, access availability between objects, availability of space for objects location, positions of objects with respect to others, and views of objects from others. These constraints are encoded as Boolean methods; if satisfied, they return “true” and if not, they return “nil”. This approach, however, eliminates the consideration of partial satisfaction of the constraints. Furthermore, all the constraints at Consite are granted the same level of importance,

which does not finely reflect the discriminating attitude of human experts in dealing with constraints.

SightPlan (Tommelein et al. 1991, Tommelein et al. 1992c, Tommelein1992) is another application of expert systems to site layout modeling for power plant sites. SightPlan is built using a specific type of computer program called blackboard architecture. Blackboard architecture allows independent knowledge sources to solve a problem cooperatively, by storing and exchanging problem elements on the blackboard, which acts as a global data structure (Hayes-Roth 1985). As in the case of other knowledge-based expert systems, SightPlan cannot include all of the expertise of the practitioners. In fact, its knowledge base contains knowledge pertaining to general spatial arrangement problems rather than domain-specific issues of site layout. However, the blackboard architecture used in SightPlan allows the users to intervene the layout decisions as another knowledge source. This feature provides the opportunity for an expert user to compensate the lack of complete domain knowledge, which is the main advantage of SightPlan over non-interactive expert systems.

Permanent and temporary facilities are represented by two-dimensional rectangles. Spatial and topographical constraints are defined among objects, but no measure has been foreseen to define preferences among constraints. Furthermore, similar to the previously mentioned models, SightPlan does not allow space reusability over time. To include the effect of time on construction sites, in a continuation study, MovePlan (Tommelein 1991, Tommelein and Zouein 1992, Tommelein et al. 1992a and 1993, Tommelein and Zouein

1993a and 1993b, Riley and Tommelein 1996, Zouein 1996) was developed as an interactive decision support tool for constructing a dynamic layout that is driven by the activity schedule.

MovePlan does not generate site layouts by itself. Rather, it is a decision support tool that verifies if the layouts created by the user for different time frames are consistent with the schedule. In MovePlan, the user is held responsible to locate the resources on site and create acceptable layouts. It is also left up to the user to determine whether there is enough space on the site to accommodate a resource at any given time. No location constraints or interaction between resources are modeled in MovePlan. The user generates time frames and identifies the resources needed for each activity, and positions resources on site. If there is not enough space on site, MovePlan issues a warning to change the schedule. The user of MovePlan is expected to generate a sequence of layouts for the span the of construction project. MovePlan provides consistency for resource positions in different layouts. It also provides the user with resource and space histograms as part of its output.

To automate the process of altering the construction schedule to resolve the spatial conflicts, MovePlan was loosely integrated with a conflict-resolver tool called ConRes (Tommelein et al. 1993, Tommelein and Zouein 1993b, Zouein 1996). When a space conflict is identified in MovePlan, ConRes is called to resolve the conflict by changing the schedule. ConRes applies several strategies to alter the schedule to fit the site constraints, while trying to keep the project duration minimal. The schedule is then sent

back to MovePlan to continue constructing layouts. MovePlan-ConRes relies on user input in constructing layouts, identifying conflicts and labeling the cause of the conflict. To automate the process of space-time tradeoff in space scheduling, another system called MoveSchedule was developed, which will be discussed shortly.

MoveCapPlan is another system developed based on MovePlan for material handling control. The system uses a bar code reader to keep track of the material and facilities on construction sites (Tommelein, I.D. 1994a, 1994b, and 1995). Similar to this system is ALYC (Automated Lay-Down Yard Control), which combines bar code and radio frequency positioning technology with a CAD system (Lundberg and Beliveau 1989). ALYC is more powerful in graphically representing objects on site. Nevertheless, both systems are designed to serve control and tracking material on sites and do not provide aid in developing layouts for construction sites.

MoveSchedule (Zouein and Tommelein 1993, Zouein and Tommelein 1994a and 1994b, Zouein 1996, Zouein and Tommelein 1999, Zouein and Tommelein 2001) is an extension of MovePlan, which constructs an activity schedule to meet the spatial constraints of the site while minimizing the duration of the project. It automatically resolves spatial conflicts by making space-time tradeoffs through a heuristic procedure. The knowledge base of the system holds project data, activity data and resource data. However, linear programming is used to find the optimum location for objects (Zouein and Tommelein 1999).

In MoveSchedule, resources are clustered in two categories: those that are associated with activities (time-dependent) and those that are not (time-independent). Time-dependent resources are again divided into two groups: consumable and productive. Productive resources (e.g. crane) set the rate of activity production (Zouein and Tommelein 1993, Zouein and Tommelein 1994a). Similar to MovePlan, MoveSchedule makes many simplifying assumptions regarding resource modeling. For example, resource representation is limited to rectangular forms; mobile resources are considered fixed in their average position; and mobilization and demobilization times are considered to be zero (Zouein and Tommelein 1994a). However, unlike MovePlan that assumes a constant space need for all the duration of resources on site, in MoveSchedule the consumption rate of resources is modeled (Zouein 1996, Zouein and Tommelein 2001).

Constraints in MoveSchedule are grouped under hard and soft constraints. Hard constraints include the none-overlap constraint (which prevents resources from overlapping), in-zone constraint (which limits the location of a facility to a predefined area), minimum/maximum distance constraint, orientation constraint (which constraints a resource to be in North/South/East/West of the other), and parallel/perpendicular constraint (which limits the orientation of a resource to that of a zone) (Zouein 1996). Soft constraints express preferences in regards to proximity and relocation weights. A proximity weight is assigned to each pair of resources to reflect the level of interaction, and hence closeness, between them. Relocation weights are used to measure the cost of relocating a facility after it is located (Zouein and Tommelein 1999).

Knowledge acquisition is often referred to as the bottleneck of expert system development (Hamiani 1987). This is due to the fact that it is hard to structure the procedure of the experts' decision-making. This makes it difficult to isolate and describe the reasoning procedure (Tommelein 1991). When developing expert systems, rules and expertise are gathered through interviews with experts. This knowledge is then interpreted by a knowledge engineer into codes that are understandable by expert systems. There is always a risk of the knowledge getting lost or misinterpreted through the way from experts to knowledge engineer and expert system. Furthermore, it is inconvenient to add new rules to the system posterior to the system implementation. The problem arises when new rules conflict with existing ones. It is essential for a feasible site layout system to be able to accommodate new rules easily, as in practice the rules and constraints change from one project to another. Furthermore, in MoveSchedule thematic information and spatial features are expressed as separate rules. These rules are not directly integrated, which makes the reasoning of the rules time consuming.

Geographic Information Systems (GIS) are unique in their ability to integrate graphical features with thematic information (Davis 1993). More importantly for layout purposes, they are capable of conducting spatial analyses. ArcSite (Cheng 1992, Cheng et al. 1992, Cheng and O'Connor 1993) is an enhanced KB Geographic Information System for solving the construction site layout problem that uses a heuristic approach to model the process of human decision-making. These heuristics were later combined with a GIS-

based cost estimating system called MaterialPlan (Cheng and Yang 2001), to locate materials on site after their quantity and size has been calculated.

The ArcSite system structure is comprised of four components: CAD system, GIS (Arc/Info), spreadsheet, and [ARC] prompt. The CAD system is needed to generate the geometric drawings including that of the site. GIS incorporates graphic features with a relational database. ArcSite's knowledge base consists of regulations, rules of thumb and expert knowledge and experience. The ability to interface with spreadsheets, databases and expert systems is an advantage that makes GIS suitable for layout purposes (Jeljeli et al.1993). However, this interface in ArcSite is seamy and the system has to move constantly from one environment to another to pursue the layout process. In order to have an efficient system, a consistent and seamless integration at the interface level is needed.

Due to the CAD system, ArcSite is able to accept and analyze various shapes for site and facilities. Further, the system provides the user with the flexibility to change facility parameters at the start of a layout design. This feature makes ArcSite more flexible in modeling than previously discussed models. The constraints defined in ArcSite are distance, adjacency, position (location of a facility compared to another one), accessibility, and space (area required for each facility). Although the process of locating each facility is automated, the system allows user interaction during the procedure of site layout generation. The system does not provide any assistance on the order of facilities entering the site and it is left up to the user to decide on this order. ArcSite is not a dynamic site layout planning system; nevertheless it considers the time factor associated

with each facility on site. If the duration of a facility on site ends, the space it occupies would be reused by the system to locate other facilities.

The process of laying out a facility in ArcSite starts by identifying the available site space. The system then searches for the optimal location for each facility by applying the constraints. To determine the optimal site for each facility, ArcSite calculates a proximity index (PI) based on travel frequency and an attract/repel factor between facilities. ArcSite also allows user interference to veto analysis results and select a preferred location among potential ones. In the evaluation phase, the output of ArcSite was different than that designed by the domain expert. This was identified as a result of over-simplification in the system design and it was determined that a more robust system was required to realistically solve the site layout problem (Cheng 1992, Cheng and O'Connor 1993).

2.3 MODELS IN RELATED DOMAINS

Site layout is closely related to other domains such as architectural layout planning, four-dimensional (4D) simulation of construction, visual space scheduling, motion planning, interference checking, and space conflict resolving. Studying these models is especially interesting for site layout modeling since some features, such as capturing and representing spatial data, can be inspiring, or directly applicable. Further, with some modifications, site layout models can be extended to include some of the aforementioned tasks. This section presents a number of such models that are closely related to site layout.

SEED is a software environment to support the early phases in building design. It addresses the issue in three categories of architectural programming, schematic architectural layout, and generation of three-dimensional configurations of physical building components, each handled by an individual module (Flemming and Woodbury 1995). SEED supports the storing and retrieval of past layouts using the case-based design. SEED-layout is one of its modules that handles the task of architectural schematic layout design. To generate layouts, SEED-layout utilizes Artificial Intelligence techniques on constraint propagation (Flemming and Chien 1995). CSL is an adaptation of this module to support the generation of construction site layouts (Choi and Flemming 1996). An interesting feature of this model is the adaptation of interdependencies between floors in a building in SEED-layout to handle the time variance in construction site layouts. For each time interval, a layout is generated similar to a stack of floors in a building. Facilities that are fixed in their position are treated as the shared objects between floors. In this way, their location is fixed once they have been placed on the layout. The objectives identified for the layout are closeness to work area, adequate work space, and access between facilities. However, only the closeness relationship is implemented. As well, in this model the facilities are modeled as rectangles for simplicity.

4D models were recognized as a viable tool to accurately visualize the construction and to facilitate communication among team members. An early work on 4D CAD modeling (Collier and Fischer 1996) visualized the construction graphically, but did not contain semantic knowledge about its building components and schedule activities. Later, in an

Interactive 4D CAD (McKinney et al. 1996, McKinney and Fischer 1997) the 4D simulation was performed through integration of a 3D graphical representation and the schedule information. In this model, the activities and building components are alert about their location and have a mutual awareness towards one another. In particular, an activity knows the components that are related to it, and conversely, a component knows which activity is going to build it. Construction Method Modeler (CMM) (Fischer and Aalami 1996, Aalami et al. 1997) is a 4D model that explicitly represents the resources required by activities and automatically generates construction schedules. In 4D WorkPlanner (Akinci and Fischer 1998) the space requirements for the activities are stored in them. As a result, the activities know when and where they are occurring, what components they are building, and how much space they occupy. Unlike generative models that produce a layout, schedule or material path that minimizes the space conflicts, 4D Workplanner is an analysis tool. In other words it analyzes an *existing* schedule for time-space conflicts. The performed tasks by this model can be summarized as simulating the construction process, identifying the time-space conflicts between activities, modifying productivity rates, and providing a feedback on the impacts of such conflicts on the existing schedule. Furthermore, a cost estimate that reflects the reduced productivity rates for each activity is calculated (Staub and Fischer 1998). This calculation is based on a crew database that consists of different crew rates. 4D WorkPlanner Time-Space Conflict Analyzer (4D TSConAn) (Akinci and Fischer 2000) is an expansion of the aforementioned model to incorporate a taxonomy of time-space conflicts. The time-space conflicts were categorized and prioritized into seven groups according to the problem they created (Akinci et al 2002).

Another research (Riley 1995, Riley and Sanvido 1997) described a representation for the spatial requirements of construction activities. The research represents each work space as a CAD object and identifies the required attributes for these objects by focusing on crew-level space requirements. The spatial requirements of the construction activities are characterized into a set of recognizable patterns such as unit work space, overhead work space, linear work space, and vertical workspace. This representation was utilized in the development of a space planning model for multistory buildings (Riley 1998). This research only considers the activities performed by enclosure and finish trades. It prompts the planner to identify the required space for each activity and define locations for each space on building floor. The model then develops a sequence of construction activities considering the space requirements of the activities and identifies the spatial conflicts. However, it does not provide a method to reflect such conflicts on the construction schedule.

ScaRC (Thabet and Beliveau 1993, Thabet and Beliveau 1994b, Thabet and Beliveau 1997) is a space scheduling model that represents the work space as CAD objects. The goal of space scheduling is to develop schedules that have minimum spatial interference between activities. It requires the user to assign the location and size of the space required for the activities. It then verifies if the required space for the activities is available at the required location and at the time specified in the schedule. If there is insufficient space for an activity, ScaRC utilizes a knowledge-based approach to resolve the conflicts (Thabet et al. 1992). In so doing, it reduces its productivity by a factor and accordingly

updates its duration time on the schedule. This research is applied for space scheduling in multistory buildings to model the activity space inside the building under construction (Thabet and Beliveau 1994a, Thabet and Beliveau 1994b). The available space on construction sites where resources are stored before they are brought to the floors is not considered in this research.

2.4 SUMMARY

There are different classes of site layout models reported in the literature. This variation stems from two main sources: the techniques utilized to analyze site space, and the assumptions made on the facilities and the constraints between them. The latter is reflected in the knowledge structure and problem representation. While the previously mentioned models address site layout with different assumptions, they suffer from one or more of the following shortcomings:

1. Limiting the optimization process to minimization of travel effort. This makes these models useful only so far as distance is concerned. In practice, the efficiency of a layout is judged by a number of other features including safety and security, which are not a function of travel effort.
2. Oversimplifying site layout representations (Cheng 1992). These models are implemented with a fixed number of temporary and permanent facilities and constraints that can be used to setup a project. Thus, they fail to represent all of

the factors that site designers use in generating a site layout. Consequently, these systems are rigid and can only handle single problem scenarios. Any change in logic or requirements calls for re-structuring these systems, rendering them impractical.

3. Neglecting user interaction. Site layout involves design issues that are dealt with differently by site planners according to their expertise and design strategies. Despite the capabilities that computerized systems provide, site planners prefer to alter decisions made by a computer system, based on their knowledge and experience. Preventing user interference, does not allow the utilization of users' experience and knowledge in designing a site layout, and more importantly, does not allow their contribution to the knowledge of the model.

These simplifications and shortcomings stem from modeling difficulties in the site layout area. Experts have different knowledge and design criteria according to their expertise and training. Most site planners work based on the individual experience, common sense, and adaptation of past layouts, which is difficult to quantify (Cheng 1992). As well, models have to extract data from various resources. This represents a communication problem in two respects: knowledge acquisition and knowledge/problem representation. Furthermore, there are no standard guidelines or methods to evaluate the quality of a layout. The objective of this research is to develop an effective site layout model designed to overcome the limitations cited above. This chapter also presented a number of models from other areas that are closely related to site layout. Studying such models is

important for site layout modeling since features such as data structure and problem representation can be inspiring for, or even applicable to site layout.

CHAPTER 3

PROBLEM REPRESENTATION FOR SITE LAYOUT MODELING

3.1 INTRODUCTION

Diversity of acceptable solutions and lack of exact rules or methods to follow have been rendered as the main problems in modeling construction site layout (Cheng and O'connor 1993). Previous surveys show that site managers design layouts based on their experience, common sense and adaptation of past layouts (Rad and James 1983). Furthermore, a range of temporary facilities can handle the same task, leaving even more options for the site managers to choose from. Several factors are considered by designers in choosing these facilities such as construction type, contract type, and project size and location (Hamiani 1987). As such, the type and composition of temporary facilities are generally regarded as project-dependent.

As discussed in the previous chapter, several computer models have been developed in recent years to support site planning. However, to date no standard tool has gained wide acceptance by industry. This has been attributed to the difference between how domain experts address the problem and the way computer-based systems represent it (Tommelein et al. 1992a). In an effort to bridge this gap, researchers have suggested the use of knowledge-level description that is more abstract and code-independent (Balkany et al. 1991). This description includes the structure, functions, and properties of the components of the model. Such description is particularly important for site layout

product models since there are few of such models that contain the domain knowledge. Further, the existing product models do not support the reuse of the knowledge and experience utilized in previous projects (Tommelein et al. 1992a). As well, these models are rigid in problem setup factors (e.g. facilities, site boundary, site conditions) and in limiting the optimization objectives to minimum travel distance (Sadeghpour et al 2004b). Therefore, guidelines for site layout model development emphasize on the importance of developing general frameworks that are capable of reusing knowledge and are transparent, interactive, and process interruptible (Tommelein et al. 1992a).

Problem representation is considered a prerequisite for computer-aided problem solving (Simon 1996) and product models aim to address this requirement. A product model presents the information structure of objects in terms of form, behavior and relation of parts and assemblies (Eastman 1999). Industry Foundation Classes (IFC), undertaken by International Alliance for Interoperability (IAI), is regarded as a major international effort to develop an integrated building model. IFC is an object oriented data model for AEC industry, which models various types of project information such as building parts and products (Frose 2002). Although a series of construction management and planning procedure has been modeled in IFC, however, the site layout area has not been tackled yet (IAI 2004). On the other hand, previous research works on site layout have mainly concentrated on the layout process and not the structure of the project setup and selection of temporary facilities. As a result, these models are rigid and comprised of a limited or fixed project setup elements such as facilities, site conditions, or layout constraints. Consequently they cannot readily be applicable to different site layout problems, but the

one they have been modeled for. Nevertheless, it is impossible to identify and embed every item and factor that is used in construction projects. Construction industry has an open nature and new methods and products are continuously introduced. The varying nature of construction techniques, as well as the unique characteristics of each project, poses constraints in mapping construction sites into computer models. Such characteristics require that computer models developed for site layout be of open architecture to allow for user interaction, not only in data processing, but in the formation of objects, methods, and constraints. As such, modeling site layout needs to be flexible and more representative of the intuitive way applied by contractors and site planners. To satisfy the characteristics of construction, such structure should be able to represent different designers' approaches, support the iterative nature of design, and be expandable to accommodate new products, methods, and technologies (Rivard and Fenves 2000).

This research presents a framework for site layout model that accounts for the aforementioned suggestions and aims to overcome the limitations cited above. It aims to address site layout modeling at product and process levels. At the product level, it deals with data structure and knowledge representation. Site layout-related elements are systematically identified and organized in a set of object libraries. This classification assists in formalizing the representation of site layout problems in a simple format. An open architecture approach is proposed to support the generation of new objects in accordance with the diverse nature of construction sites. Thus, it facilitates user contribution to the expansion and refinement of the data and knowledge of the model, which in turn, enriches the model's capabilities. Furthermore, it assists in recycling the

domain knowledge used in current projects for later use on future projects. At the process level, a geometric approach for analyzing the site space is introduced. This approach resembles the human reasoning process and is aimed to address the mismatch, cited in the literature, between the practitioner's approach to the problem and that of existing models. The geometric analysis of site space is carried out under a set of constraints assigned to facilities. In an effort to improve user recognition and enhance visualization, a graphical representation of constraint satisfaction is proposed. This representation displays constraint satisfaction in a simple format that makes the search process easily understandable and visually traceable for planners. As well, it facilitates the assessment and evaluation of identified locations for objects.

3.2 DATA STRUCTURE

The site layout problem is modeled using a set of objects, along with their functionality and inter-relationships. The process of site layout is formulated to place a set of *construction objects* on site, while respecting a set of *existing objects on site* and satisfying a set of locating *constraints* defined by the user. As such, the site layout elements are clustered into three groups of objects: 1) *site objects*; 2) *construction objects*; and 3) *constraint objects* (Sadeghpour et al. 2002, Sadeghpour et al. 2003).

Object-based concepts have been utilized to represent the three tiers of objects. Object-based approach strongly promotes the formalism of data typing and encapsulation of information. Contrary to the object-oriented approach, no explicit inheritance scheme is imposed in the object-based approach (Zamanian 1992). The proposed tiers are

implemented as object classes, encapsulating their respective attributes. These attributes include the geometrical and non-geometrical data and knowledge. Introducing a formal structure for the objects enables the creation of new objects, in an effort to be compliant with the changing nature of construction. The three aforementioned tiers of objects are described in more details below. In the remaining of this thesis, except for chapter 5, the terms “property” and “attribute” are used interchangeably.

3.2.1 Site Objects

Site objects include the site boundary and objects that reside on site before the commencement of construction, and hence have a known location on site. It is important to include site objects in the modeling since they affect the search process for the location of construction objects, and hence the final layout. Some examples of site objects are trees, existing buildings, specially marked areas on the site such as unavailable, unsafe, or hazardous, water ponds, life lines such as sources of electricity, water, or phone lines, and underground activities such as excavation or piping. Site objects often exist on site permanently; however their duration on site can be specified if needed. In spite of their impact on site layout, very few research works have modeled site objects (e.g. Lundberg et al. 1989). Site objects play two main roles in site layout: 1) they occupy space on site, so the area they occupy is deducted from the total site land; 2) their topological relations with construction objects define the constraining rules and hence affect the final layout of construction sites. The conceptual design of site objects includes a set of properties that can be clustered into three groups of layout, geometrical, and graphical. Layout properties are non-geometrical information concerning site layout issues such as duration

of an object on site. Geometrical properties include information such as area of an object and its location on site. Graphical properties contain information regarding the appearance of the object.

Table 3.1. Properties of site objects

	Property	Description	Example
Layout Properties	ID	A unique ID assigned to each site object upon its creation	Site005
	Name	Name of the site object	Water pond
	Arrival time	Date of object's arrival on site. For site objects this date is usually the beginning of project	14/02/2004
	Departure time	Date in which the object is removed from site. For site objects this date is usually the end of project	14/02/2005
	Mobility	Indicates if the object is mobile or stationary	Stationary
	Movability	Indicates if it is possible/acceptable to change the location of object during the project (Yes/No). The location of site objects is usually fixed.	No
	Cost of relocation	For movable objects, this property indicates the desirability to change their location during project	N/A
	Containability	Indicates if the object can be used later to contain other objects inside (Yes/No)	No
Geometrical Properties	Area	Footprint area of site object	12.00 m ²
	Centroid	Coordinates of geometric centroid of footprint	(1.5, 2, 0)
	Perimeter	Perimeter of footprint	17 m
	Location	Coordinates of predetermined location of site object	(24, 15, 0)
	Height	Highest point of object	1.3 m
	Orientation	The angle by which the object is rotated when located on site	458
Graphical Properties	Color	Color in which the geometry of object appears	Red
	Layer	Drawing layer name in which the object is located	WaterPound
	Visibility	If the object is visible or hidden on screen (Yes/No)	Yes

Table 3.1 contains the list of site object properties at the conceptual design level. At this level, the effort was to identify a general list of attributes that are necessary for the task of site layout, regardless of the data processing approach, and implementation techniques and environment. Clearly, the locating approach and design of the model will require some modifications to this list. As well, at the implementation level, the capabilities and features of the selected environment imposes justifications on the required properties for objects.

3.2.2 Construction Objects

Construction objects enter the site at the commencement and during the course of construction. The objective of site layout is to find the optimum or near optimum locations for these objects. Construction objects address a range of items that are diverse in nature including equipment, material, temporary support facilities, buildings, lay down areas, working areas, and generally objects that occupy space and have to be located on site. Appendix I contains a list of most commonly used construction and site objects. Similar to site objects, construction objects are as well represented by object class. The structure of a construction object is similar to that of a site object with a set of layout, geometrical and graphical attributes. The properties of construction objects at conceptual design level are summarized in Table 2.

Construction objects share most of their attributes with site objects. The main difference between the two is that unlike site objects, the location of construction objects is not known at the commencement of the site layout process, but is to be defined during that process. Further, construction objects have a set of layout properties regarding their

sizing on the site. Some construction objects such as materials are delivered in packages, boxes, or other units. So when being located on site, they can be distributed in separate

Table 3.2. Properties of construction objects

	Property	Description	Example
Layout Properties	ID	A unique ID assigned to each construction object upon its creation	Const005
	Name	Name of the construction object	Crane
	Arrival time	Date of object's arrival on site. For site objects this date is usually the beginning of project	14/05/2004
	Departure time	Date in which the object is removed from site. For site objects this date is usually the end of project	01/01/2005
	Mobility	Indicates if the object is mobile or stationary	Stationary
	Movability	Indicates if it is possible/acceptable to change the location of object during the project (Yes/No)	Yes
	Cost of relocation	For movable objects, this property indicates the desirability to change their location during project	100
	Containability	Indicates if the object can be used later to contain other objects inside (Yes/No)	No
	Flexibility	Indicates the flexibility of object's shape (flexible/ sizable/ rigid)	Rigid
	Smallest unit	If the construction is sizable, this property indicates the dimension of its smallest units	N/A
Geometrical Properties	Area	Footprint area of site object	12.00 m ²
	Centroid	Coordinates of geometric centroid of footprint	(1.5, 2, 0)
	Perimeter	Perimeter of footprint	17 m
	Location	Coordinated of the location found for construction object. The value of this property is found in site layout process.	(24, 15, 0)
	Height	Highest point of object	35 m
	Orientation	The angle by which the object is rotated when located on site	08
Graphical Properties	Color	Color in which the geometry of object appears	Blue
	Layer	Drawing layer name in which the object is located	Crane 1
	Visible	If the object is visible or hidden on screen (Yes/No)	Yes

locations, if needed (e.g. bricks). The dimension of the smallest unit is needed to decide the size of split locations. Bulk material such as sand, nails, and fasteners, are flexible in shape, so there is no sizing constraint when dividing them into different locations. Objects such as equipments are rigid in size and cannot be divided. This is reflected under *flexibility* property, taking values of flexible, sizable, or rigid. This property also determines whether overlapping with other objects is allowed or not. The defining boundary of rigid objects cannot be overlapped with other objects. For flexible and sizable objects overlapping is allowed, since the geometry representing the area they occupy is resizable. *Containability* indicates if the object, once located on site, can be used to house other objects inside its boundaries. Examples of this are the storage trailer or lay down areas.

3.2.3 Constraint Objects

The designers' approach is best understood in terms of constraints (Hamiani 1987). As such, expressing location constraints and interaction between objects is considered the key to developing site layout models (Zouein 1996). In this research, the process of finding optimum location for a construction object is carried out under a set of rules. These rules are mapped into a set of related objects referred to as constraint objects. Different constraint objects can be defined based on the objectives of the layout. As a result, they are not limited to minimum travel distance, but include constraints to express other objectives such as safety and security. Appendix I includes a set of locating rules for site layout extracted from communications with site managers and construction safety code of Quebec province (ASP Construction 2001). Constraint objects contain rules

defined to represent the relationships among site objects and construction objects; which when applied, satisfy the layout planning objectives. These locating rules, along with a relative weight assigned to them, constitute the knowledge of the model.

Devising a general structure for constraint objects is more challenging than that for site and construction objects, since they have neither geometry nor physical properties in the same sense. The challenge is in finding a method, capable of mapping constraints as defined by users in natural language. In an effort to address this problem, a generic pattern in the constraints expressed for locating objects in site layout was recognized. Consider the following example:

“Parking should be located as close as possible to temporary office.”

A

C

B

Three elements are noticeable in this constraint statement; A) the object for which a location on site is searched and for which the constraint is being defined, B) the object that constrains the location of object A, and C) the spatial relationship by which object B constrains the location of object A. Therefore constraints can be expressed, or re-structured, in the format of “A should be C to B”. As such, the structure of each constraint is designed to be composed of three elements: 1) *constrained element* for which the constraint object is being defined; 2) a spatial *relationship*, such as “as close as possible to”, “west of”, “visible from”, and “within 5 meters of”; and 3) *constraining element*, which delimits the location of the *constrained element*. For example to apply security constraint when locating a stack of electrical appliances, they can be defined to

be ‘visible from’ and ‘close to’ guardhouse. Alternatively they can be locked ‘inside’ a trailer or a warehouse.

Figure 3.1 illustrates the schematic structure of constraint objects. The constrained element is always a construction object. Constraining element, however, can be either another construction object, or a site object. As an example, a safety constraint indicates that the explosives should not be kept close to offices. Offices are constraining the location of explosives with the spatial relation of “far from”. When the constraining element is a construction object, like a trailer office, the constraint object is called (C-C). On the other hand, if the constraining element is a site object, like an existing office building on site, the constraint object is classified as (C-S) (Figure 3.1). The developed structure facilitates the creation of new constraint objects by providing a method to ‘*mix and match*’ the three consisting elements, upon the availability of a set of each.

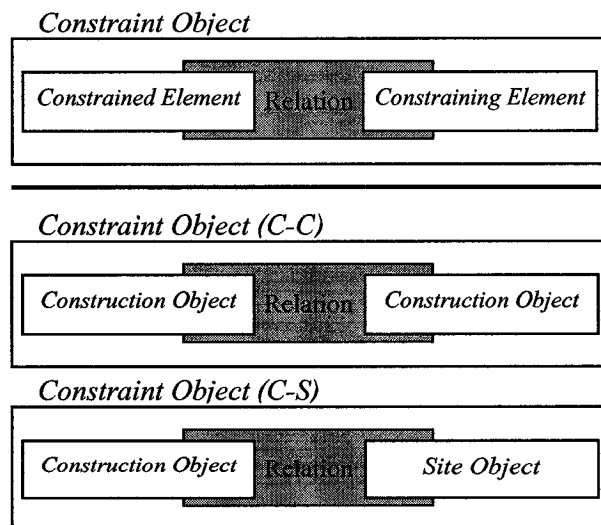


Figure 3.1. The structure of constraint object

Constraint objects are as well represented by object class. The attributes of a constraint objects are shown in Table 3.3. Since it is not always possible to satisfy all the constraints assigned to one construction object at the same time, each constraint object is assigned a priority weight. This weight is a number between 0 and 100 and indicates the importance of the constraint object with respect to others. In this dissertation, the terms ‘constraint’ and ‘constraint object’ are used interchangeably.

Table 3.3. Properties of constraint objects

Property	Description	Example
ID	A unique ID assigned to each constraint object upon its creation	Consr005
Name	Name of the constraint	Safety 1
Constrained object	The construction object for which this constraint has been defined	Explosives
Relation	A spatial relationship that is desired to exist between constrained and constraining objects	Far from
Constraining object	A construction or site object that constrains the location of the construction object	Office
Weight	Indicates the importance and priority of the constraint object over other constraints	90

3.3 RELATIONSHIPS AMONG OBJECTS

Figure 3.2 illustrates the relationships among the three tiers of objects, site layout project, and spatial relationship- referred to as site layout entities- in UML notation. Each project *has* a set of site, construction, and constraint objects. Objects can be shared between different projects. Constraint objects are complex objects in that they have other objects in their structure; they consist of site object, construction object, and relationship. The three elements can be shared between various constraint objects. A C-S constraint has one site object and one constraint object in its structure, where a C-C constraint is comprised of two constraint objects.

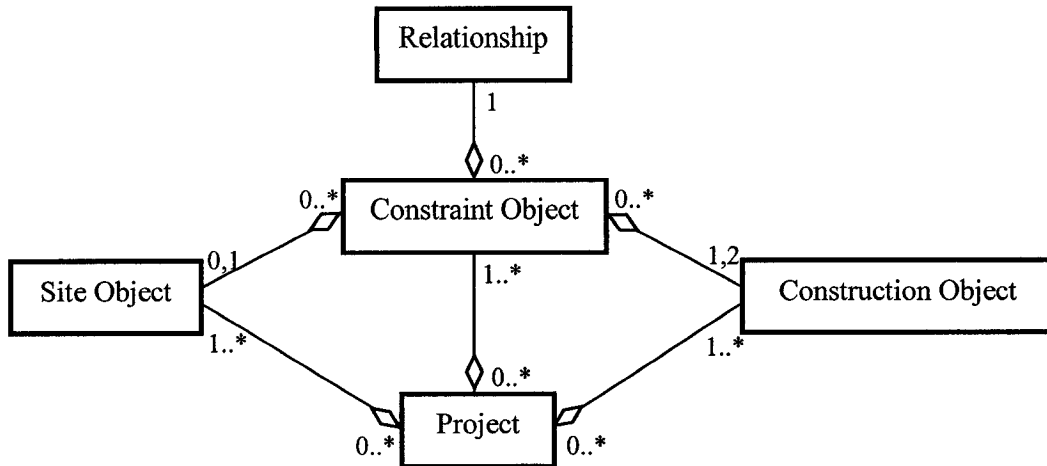


Figure 3.2. Conceptual relations among site layout entities in UML notation

3.4 OPEN ARCHITECTURE FOR SITE LAYOUT MODELING

Open architecture utilized in this research calls for the formation of general categories that host a number of objects relevant to the problem being modeled. Selection of categories is based on the intuition of the model designer respecting two principles: 1) within each category, the entities share the same attributes; and 2) the selected categories are adequate to describe and present the project being modeled. The proposed model, described in next chapter, applies these principles of open architecture to site layout problem.

The model maintains libraries for the three tiers of objects described previously: “site library”, “construction library”, and “constraint library”. Each library is contains the respective objects. When starting a new project, required setting objects have to be defined. The term *define* denotes two functions in the proposed model: *selecting* objects from the respective library, and *creating* new objects. Each of the site library,

construction library, and constraint library, offers a set of respective objects from which the user can *select* objects suited to represent the project at hand. If a desired object is not available in the model's libraries, the model provides tools to *create* it. As such, the model applies open architecture concepts for setting up a project. Once a new object is created, it is automatically added to the corresponding library. This eliminates duplication and redefinition of objects for different projects. It also supports the expansion and enrichment of the model's libraries, and more importantly, gradually customizes it according to design needs and preferences of its users.

The open-architecture design of the proposed model allows it to interact with users at both expert and novice levels. The first level provides the domain expert with tools to enrich the libraries. This allows planners to apply their individual problem solving strategies, and thus, directly contribute to the knowledge contained in the libraries. This feature eliminates the traditional need of a knowledge engineer for acquiring and structuring the extracted knowledge, and hence decreases the risk of misinterpretation and incomplete acquisition of relevant knowledge. Based on the model's status of knowledge, the project module provides less-experienced site planners with a decision support for defining the requirements of a site layout project.

3.5 GEOMETRICAL REASONING FOR SITE SPACE ANALYSIS

Studies show that the use of visualization aids substantially in restructuring the knowledge and facilitating comprehension of the subject (Casakin and Goldschmidt 1999). Benefiting from the potential of visualization, a geometric reasoning search

methodology has been developed for the site space analysis (Sadeghpour et al. 2004c). In this method, a geometric interpretation for each spatial relationship is developed and presented in graphics. For each constraint the site is divided into segments, representing a different degree of satisfaction for that constraint. The discretization method differs according to the type of spatial relationship used to define the constraint (Sadeghpour et al 2004a).

Spatial characteristics of information are those that describe i) the whereabouts of phenomena; using locations consisting of reference, positions, spatial units and spatial relationships; ii) form of phenomena; using qualitative or quantitative descriptions of shape and structure; and iii) associations and interactions among phenomena. Descriptions of locations can be expressed nominally or metrically. Metric locations define a position relative to a referenced location by using spatial relationships. Spatial relationships can be categorized into *topological*, *proximal*, and *directional*. Topological relationships are orientation-independent and can be generally reduced to equivalence, partial equivalence (overlap, cross), containment (inside, outside), and adjacency (adjacent, disjoint). Directional relationships are orientation-dependent and include cardinal directions and their combinations, metric description of angle of azimuth, and relative orientation (in front, behind, above, below). Proximity relationships describe distance between two objects, either quantitatively, or qualitatively (close to, far from) (Jones 1997).

For the purpose of site layout, six types of spatial relationships are adopted in this research: closeness, distance, adjacency, containment, orientation, and visibility. Visibility is not considered a spatial relation in the literature; however, it will be discussed that it can be defined in terms of proximity. These relationships are obviously not exhaustive, but are sufficient for defining most of the cases. Table 3.4 summarizes the relationships considered in this research. Based on their satisfaction scheme, the relationships are divided into two groups: 1) linear relationships, and 2) quasi relationships. Linear relationships represent a gradual transformation in the satisfaction score for each site segment, where as quasi relationships divide site in two segments and demonstrate a strict preference of one segment over the other.

Table 3.4. Classification of the considered relationships

	Relation Type	Relationship	Category
Linear relations	Closeness	Close to (Polar) Far from (Polar)	Proximity
		Close to (Linear) Far from (Linear)	
Quasi-relations	Distance	Within x distance Not within x distance	Proximity
	Adjacency	Adjacent Disjoint	Topological
	Containment	Inside Outside	Topological
	Orientation	East of, West of North of, South of	Directional
	Visibility	Visible from	Proximity

3.5.1 Geometric representation of constraints

Brans et al. (1986) used six mathematical functions to represent the satisfaction scheme when ranking items in a multi-criteria analysis. Two of these functions, namely *staircase* function and *step* function, are utilized here to graphically represent the relations defined among objects. The staircase function (Figure 3.3) takes a constant value for each interval unit of the variable being considered. As such, the value of the function changes gradually in a step-wise manner. Therefore, step functions are suitable to represent linear relations. Depending on the slope of staircase function (i.e. descending or ascending), the function value rises or falls by increment h for each interval v . The function takes a value between 0 and 1, represented in the y axis in Figure 3.3. The horizontal axis represents the variable *distance*, limited between 0 and L . As such, the ascending staircase functions shown in Figure 3.3a can be expressed as:

$$CS = h \cdot \left\lfloor \frac{x}{v} \right\rfloor \quad 3.1$$

In which x is the *distance* variable, v is the interval distance, and the brackets denote the floor function which truncates the decimal portion of the argument. Similarly, the descending staircase function (Figure 3.3b) can be expressed as:

$$CS = h \cdot \left\lfloor \frac{L - x}{v} \right\rfloor \quad 3.2$$

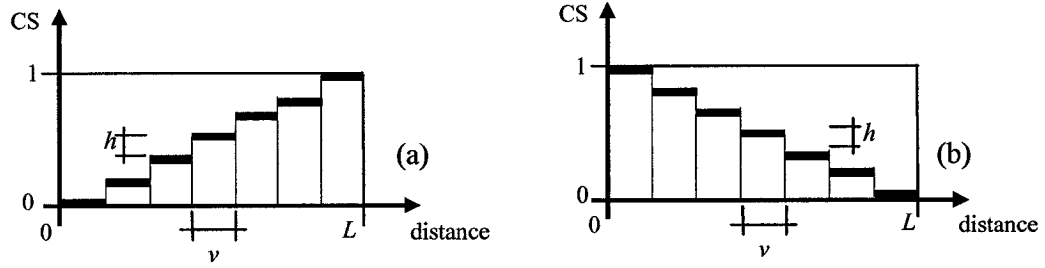


Figure 3.3. Staircase Function (a) Ascending (b) Descending

The step function (Figure 3.4) is binary, taking a value of 0 or 1, to demonstrate a strict preference of one condition over the other. The value of the function changes only when the distance from the constraining element exceeds the threshold value d . If q is the *distance* variable, for falling step (Figure 3.4a) this function can be expressed as:

$$CS = \begin{cases} 1 & \text{if } 0 < q < d \\ 0 & \text{if } d < q < L \end{cases} \quad 3.3$$

In which L is the largest allowed distance. And rising step (Figure 3.4b) is written as:

$$CS = \begin{cases} 0 & \text{if } 0 < q < d \\ 1 & \text{if } d < q < L \end{cases} \quad 3.4$$

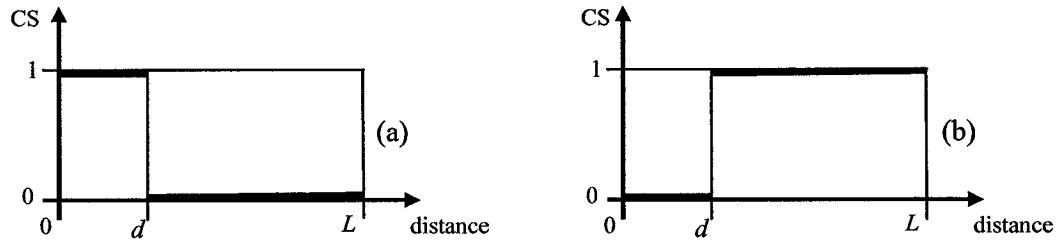


Figure 3.4. Step function (a) Falling step (b) Rising step

Due to the binary nature of the step functions, in this research they are used to present the quasi-relations, e.g. adjacency, orientation, distance, containment, and visibility. Equations 3.3 and 3.4 are used to represent the first four relations, with d and q being redefined differently for each relation. However a separate binary function is defined to represent the visibility relation.

3.5.1.1 Closeness Relations

Closeness relations are linear, for which the level of constraint satisfaction (CS) changes gradually as the distance from the constraining element increases. Since the distance is discretized into intervals, the staircase function is suitable to express the gradual change in a step-wise manner. As the distance from the constraining element increases, the CS value for the “close to” relation decreases (Figure 3.3b), and conversely for the “far from” relation, this value increases (Figure 3.3a). Equations 3.1 and 3.2 mathematically express the “far from” and “close to” relations respectively, in which L is the longest dimension of site, and v denotes the distance intervals.

To graphically represent the polar closeness relationship, the site is divided into rings by circles circumscribing the constraining element (Figure 3.5a). The number of rings is calculated as:

$$N = \frac{L - v}{v} \quad 3.5$$

or

$$N = \frac{l}{v} - 1 \quad 3.6$$

in which v denotes the offset distance between circles, or the width of the rings, and l indicates the longest distance between the centroid of the constraining element and the site perimeter. The offset distance v can vary according to the desired precision. It is assumed that the area inside each ring has the same distance from the constraining element. The constraint satisfaction (CS) score for each ring is measured as a factor of its distance from the constraining element, with 1 showing the highest satisfaction and 0 the lowest. For the “close to” relation, the closest ring to the constraining element gets the highest score, and the farthest ring gets the least. Conversely for the “far from” relation, the closest ring to the constraining element is the least desirable and the farthest is the most desirable. Accordingly, each ring is assigned a CS value for the closeness relation. The CS value for any location on site is assigned based on the ring it is located in. Thus, the rings aid in visually recognizing the satisfaction degree of different locations on site for polar closeness constrains.

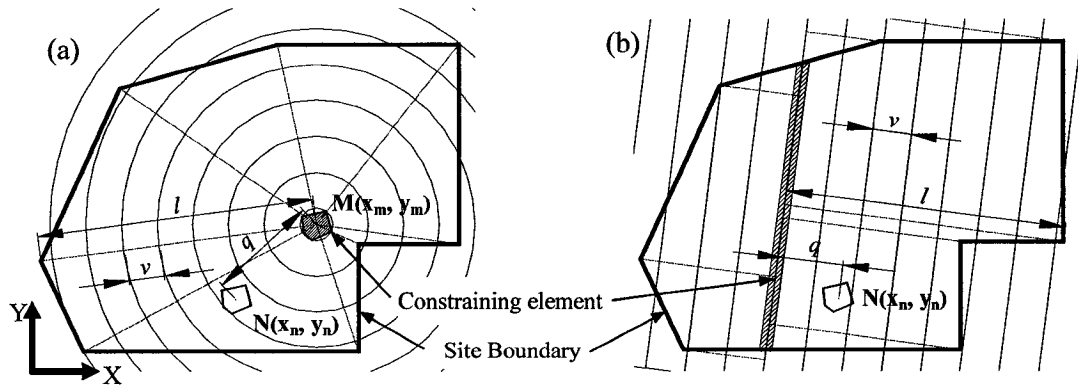


Figure 3.5. Graphical representation for closeness relation (a) Polar closeness (b) Linear closeness

Linear closeness relations are defined to express the proximity to constraining objects that are linear in shape, such as roads, walls, or site edges. Here, instead of rings, the site is divided into stripes parallel to the constraining element (Figure 3.5b). Similarly, the maximum number of offset stripes on each side is calculated using Equation 3.6, in which v denotes the offset distance, or the width of the stripes, and l indicates the longest distance between the constraining element and the site perimeter.

3.5.1.2 Distance Relations

Distance relations express the preference for an object to be located within, or not within a certain distance from another object on site. Step functions are used to express this relation mathematically. Consider a constrained element that is desired to be “within d distance” from a constraining element. Equation 3.3 describes the CS value, if q is described as the distance between the constraining element and the constrained element, or:

$$q = \sqrt{(x_m - x_n)^2 + (y_m - y_n)^2} \quad 3.7$$

in which (x_m, y_m) and (x_n, y_n) represent the coordinates of the constraining element and the constrained element, respectively. Figure 3.6 shows the graphical representation for distance relation. The area within the circle A, with a diameter of d , gets the highest satisfaction score, and region B with distance larger than d from the constraining element, gets null. In other words, if R is the location of the constrained element:

$$CS = \begin{cases} 1 & \text{if } R \in A \\ 0 & \text{if } R \in B \end{cases} \quad 3.8$$

Conversely for the “not within d distance” relation, the area inside the circle A in Figure 3.6 gets null, where the region B gets 1 for the score:

$$CS = \begin{cases} 1 & \text{if } R \in B \\ 0 & \text{if } R \in A \end{cases} \quad 3.9$$

As it can be inferred from Figure 3.6, when the constraint is presented graphically, it is easier to visualize the degree of satisfaction based on Equations 3.8 and 3.9 than calculating the distance q and using Equations 3.1 and 3.2.

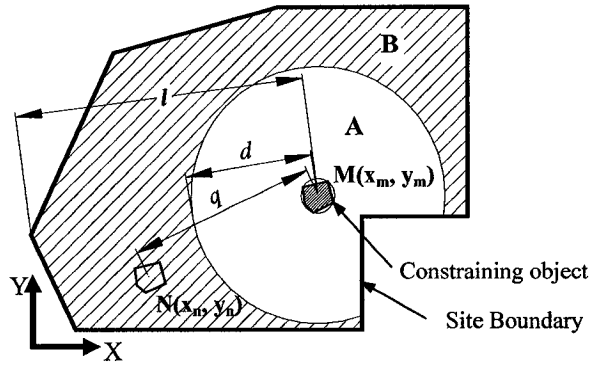


Figure 3.6. Graphical representation for distance relation

3.5.1.3 Adjacency Relations

In the proposed method, two objects are considered adjacent if the distance between their centroid is half of the sum of their largest respective dimension, represented by d .

$$d = \frac{m}{2} + \frac{n}{2} \quad 3.10$$

where m is the largest dimension of the constraining element M; and n is the largest dimension of the constrained element N. Clearly, the definition of adjacency adopted here is different from one that defines it by having an overlapping edge between two objects.

Although based on the definition provided here, adjacent objects may be loosely separated, the precision is sufficient for the purpose intended here. If the smallest dimension of M and N are denoted as m' and n' respectively, the maximum distance that objects can take from each other in the aforementioned definition of adjacency can be measured as:

$$e = 1/2 \cdot [(m - m') + (n - n')] \quad 3.11$$

which is negligible. If q describes the distance between two objects as in Equation 3.7, Equation 3.3 can be used to mathematically describe the status of adjacency between two objects. Geometrically however, the adjacency status is defined with membership function in Equation 3.8, where A represents a circle with a radius of d and its center is the centroid of the constraining element. Region B consists of the remaining site area (Figure 3.7). Conversely, Equation 3.9 verifies the satisfaction value for the disjoint relation (e.g. if two objects are not located next to each other).

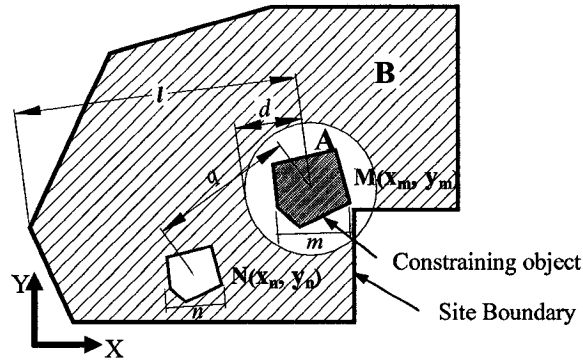


Figure 3.7. Graphical representation for adjacency relation

3.5.1.4 Orientation Relations

Orientation relations are used to describe the geographical positioning of an object with respect to another, and are commonly utilized in construction sites. In this section the cardinal directions, i.e. “east of”, “west of” “north of” and “south of”, are discussed. Nevertheless the orientation relations can be used to describe intercardinal points such as “north-west of” or relative positions expressed by the angle of azimuth. Consider the following example: “constrained element should be located *west of* constraining element”. If the north orientation is aligned with the Y axes as shown on Figure 3.8, and q represents the x coordinate of the constrained element ($q = x_n$), and l is the longest dimension of site along x axis, Equation 3.1 can be used to mathematically measure the constraint satisfaction (CS) for the “west of” relation.

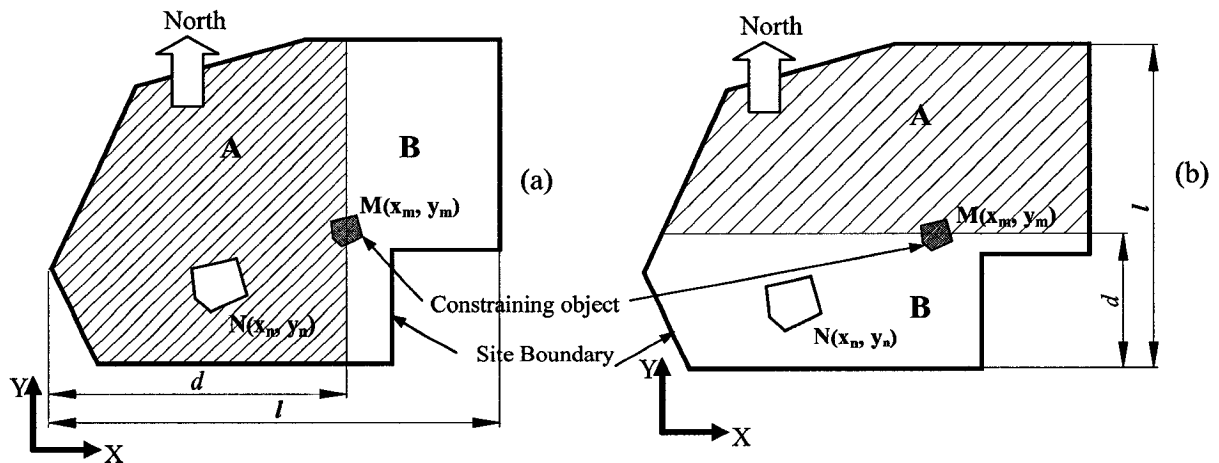


Figure 3.8. Graphical representation for orientation relation (a) East/West relations (b) North/South relations

To geometrically express this relation, the site is divided into two polygons as shown in Figure 3.8a. The division line is parallel to the North-South axis and passes through the

centroid of the constraining element. The constraint is satisfied if the constrained element is located in polygon A, and not satisfied if in polygon B, as expressed in Equation 3.8. Conversely, constraint satisfaction for the “east of” relationship is decided as indicated by Equation 3.9. Similarly relations “north of” and “south of” can be graphically expressed as shown in Figure 3.8b.

3.5.1.5 Containment Relations

Containment relations are used to express the preference of locating an object “inside” or “outside” another object, such as staging areas or areas marked hazardous. Equation 3.7 measures the distance q between two objects. Equation 3.3 can be used to mathematically describe if the constrained element is “inside” the constraining element. Conversely, Equation 3.4 is used for the “outside” relation. In both equations d is the longest distance from the centroid of the constraining element to its perimeter. Figure 3.9 illustrates the graphical representation for the containment relations. The “inside” and “outside” relations are defined by the geometry of the constraining element; where region A is the area delimited by its boundary, and B is obtained by subtracting region A from the site boundary. Geometrically, the containment is defined with membership functions in Equation 3.8 and 3.9, expressing the “inside” and “outside” relations, respectively.

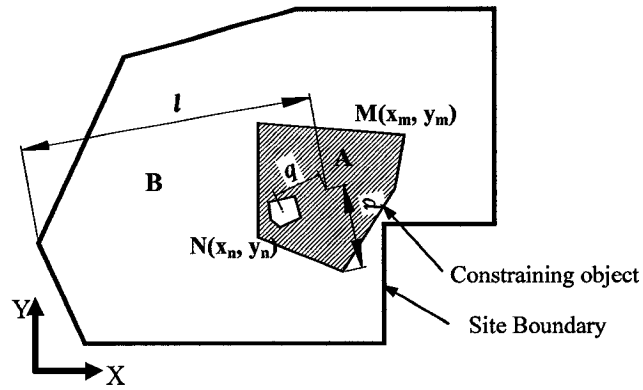


Figure 3.9. Graphical representation for containment relations

3.5.1.6 Visibility Relation

Visibility is not a traditional spatial relation, and hence it is different from the relations described earlier. However it can be expressed by means of relative positions and distances between constraining and constrained elements. When examining visibility between two objects, twelve situations can occur depending on their relative height. If M , N , and P represent the constraining element, constrained element, and obstacle, respectively, and h_M , h_N , and h_P represent their respective heights, the conditions can be clustered into the following three groups. Note that h_M indicates the vision height of the constraining element, where h_N , and h_P indicate the actual height of objects.

- i. M , N , and P have three different heights. In this case six situations (i.e. $3! = 6$) can happen:
 1. $h_M > h_P > h_N$; where visibility is conditional.
 2. $h_M > h_N > h_P$; where N is visible from M .
 3. $h_P > h_M > h_N$; where N is not visible from M .
 4. $h_P > h_N > h_M$; where N is not visible from M .

5. $h_N > h_M > h_P$; where N is visible from M.
 6. $h_N > h_P > h_M$; where visibility is conditional.
- ii. Between M, N, and P, two objects have the same height and the third is taller. In this case 3 (i.e. 3x1x1) situations can occur:
7. $h_M = h_P < h_N$; where N is visible from M.
 8. $h_M = h_N < h_P$; where N is not visible from M.
 9. $h_N = h_P < h_M$; where N is visible from M.
- iii. Between M, N, and P, two objects have the same height and the third is shorter. In this case 3 (i.e. 3x1x1) situations can occur:
10. $h_M = h_P > h_N$; where N is not visible from M.
 11. $h_M = h_N > h_P$; where N is visible from M.
 12. $h_N = h_P > h_M$; where N is not visible from M.

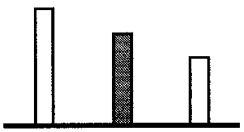
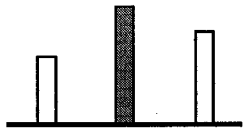
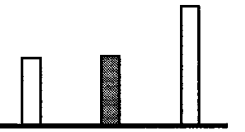
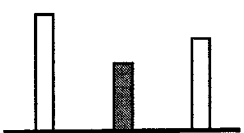
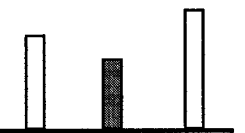
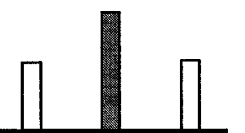
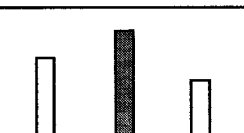
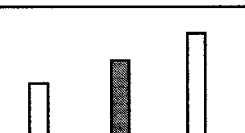
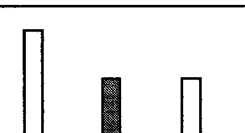
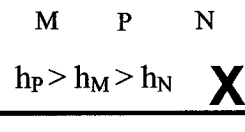
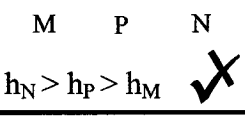
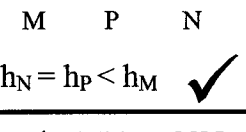
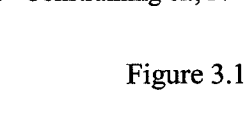
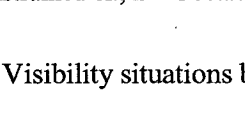
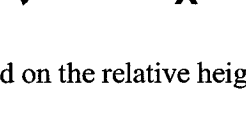
Figure 3.10 depicts the aforementioned scenarios. The following conclusions can be inferred:

If $(h_P \leq h_N)$ and $(h_P \leq h_M)$ then N is visible from M.

If (h_P / h_N) and (h_P / h_M) then N is not visible from M.

If $(h_M > h_P > h_N)$ or $(h_N > h_P > h_M)$ then the visibility of N from M is conditional.

This conditional scenario is discussed below.

i		ii		iii	
	$h_M > h_P > h_N$ ✓✗		$h_P > h_N > h_M$ ✗		$h_M = h_P > h_N$ ✗
	$h_M > h_N > h_P$ ✓		$h_N > h_M > h_P$ ✓		$h_M = h_N > h_P$ ✓
	$h_P > h_M > h_N$ ✗		$h_N > h_P > h_M$ ✓✗		$h_N = h_P > h_M$ ✗
	$h_M = h_P < h_N$ ✓		$h_M = h_N < h_P$ ✗		$h_M = h_N < h_P$ ✗
	$h_N = h_P < h_M$ ✓		$h_N = h_P < h_M$ ✓		$h_N = h_P < h_M$ ✓

M= Constraining el.; N= Constrained el.; P= Obstacle ✓ Visible ✗ Not visible ✓✗ Conditional

Figure 3.10. Visibility situations based on the relative height of objects

In Figures 3.11a and 3.11b, d represents the optimal distance between constraining and constrained elements. It can be expressed as:

$$d = \frac{(h_M - h_N) \cdot d_p}{(h_M - h_P)} \quad 3.12$$

Where h_N , h_P , and h_M respectively represent the heights of N, P, and M; and d_p is the distance between the obstacle and the constraining element. For the first case where $h_M > h_P > h_N$ (Figure 3.11a), d is the minimum distance between N and M so that the visibility can be maintained. Figure 3.11c shows the blind zone for the constraining

$$CS = \begin{cases} 0 & \text{if } (d_p < q < d) \cap (a_1 < a < a_2) \\ 1 & \text{if } (d < q < l) \cup (a < a_1) \cup (a_2 < a) \end{cases} \quad 3.13$$

66

Conversely for the second case, if $h_N > h_P > h_M$, d represents the maximum distance between N and M to maintain the visibility (Figure 3.11b). The blind zone A in Figure 3.11d is restricted between ring $d-l$ and the boundary lines of the field of vision. The constraint satisfaction for the visibility relation can similarly be written as:

$$CS = \begin{cases} 0 & \text{if } (d < q < l) \cap (a_1 < a < a_2) \\ 1 & \text{if } (0 < q < d) \cup (a < a_1) \cup (a_2 < a) \end{cases} \quad 3.14$$

Graphical determination of visibility is carried out by means of drawing vision line from the constraining element M to constrained element N. If this line intersects an object on site, the height of the three objects is compared to decide the visibility situation (Figure 3.10). If the visibility is conditional, then d is measured and the blind zone (region A) is generated. The margins of region A are defined by lines that emanate from the constraining element M and are tangent to obstacle P; and circles with radii of d and d_p , or d and l , depending on the case. The remaining of site forms region B, in which if the constrained element is located, it will be visible from the constraining element. Equation 3.9 expresses the CS value for visibility relation with a membership function. It should be noted that the planar dimensions (width and length) of the constrained element are not considered in evaluating visibility.

3.5.1.7 Compound Relations

Once the basic relations are identified along with their geometrical representation, new layout constraints can be defined, or re-structured using the basic relations. For example to consider *security* constraint for valuable objects, they can be locked “*inside*” a storage

container; be “*visible from*” guard house; or located “*in*” specially marked areas of site. Similarly compound relations can be achieved by combining two or more basic relations. Consider the following constraint example: “*The constrained element should be located the closest within x meters to the constraining element.*” This constraint can be analyzed into two constraints, each defined by a basic relation:

- i. The constrained element should be *within* x meters from the constraining element.
- ii. The constrained element should be as *close* as possible to the constraining element.

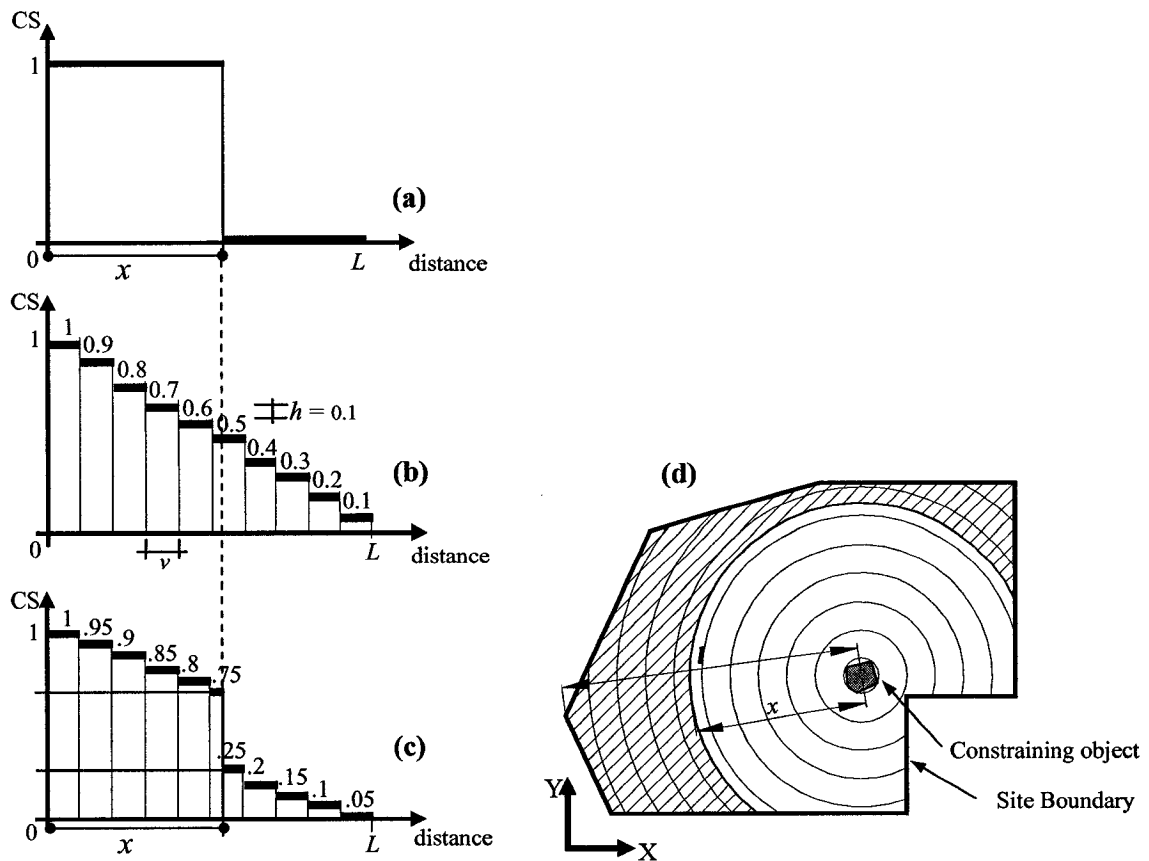


Figure 3.12. Compound relation (a) Descending step function for distance relation (b) Descending stair function for closeness relation (c) Compound function (d) Graphical representation for the compound relation

Figure 3.12c shows the graph for the aforementioned constraint composed of falling step (Figure 3.12a) and descending staircase (Figure 3.12b) functions and Figure 3.12d is the graphical representation for the compound relation from the overlap of the two. The satisfaction scheme for the new constraint is as well scaled from 0 to 1.

3.5.2 Site Space Analysis Considering Multiple Constraints

The process of analyzing site space to place objects on site is carried out through a geometric reasoning search (Sadeghpour et al. 2004c). The developed methodology aids the planners to search for optimum or near-optimum locations for objects on site using the described geometric representation of constraints. When analyzing site space for locating a construction object, all the constraints defined for that object are considered. This includes all the constraint objects whose *constrained element* is the construction object at hand. For each constraint defined for the construction object at hand, the site is divided into regions. The discretization method differs according to the type of spatial *relationship* used to define the constraint as discussed in the previous section. Each segment takes a constraint satisfaction (CS) value, considering the weight assigned to that constraint.

When analyzing site space, the geometric representation for all the constraint objects associated with the construction object in hand are generated and overlapped on one site boundary. For example Figure 3.13 shows the site segments generated for three constraints indicating closeness to site objects 1, 2 and 3. Accordingly, the site is divided into smaller and smaller segments resulting from intersections of the geometries

generated for the graphical representation of the three closeness relationships. Figure 3.14 shows site segments before and after the intersections of closeness rings from Figure 3.13. This process divides the site into a set of irregular segments that are neither of equal size nor of same shape. Considering the fact that the intersection of two areas contains the attributes of both, the satisfaction score for the area of intersection is calculated as the sum of the scores associated with the two intersecting areas. As such, the *utility score* of each site segment is expressed as:

$$U = \sum_{i=1}^n S_i \cdot W_i \quad 3.15$$

where S_i is the satisfaction for the i th constraint; and W_i is weight assigned to the i th constraint.

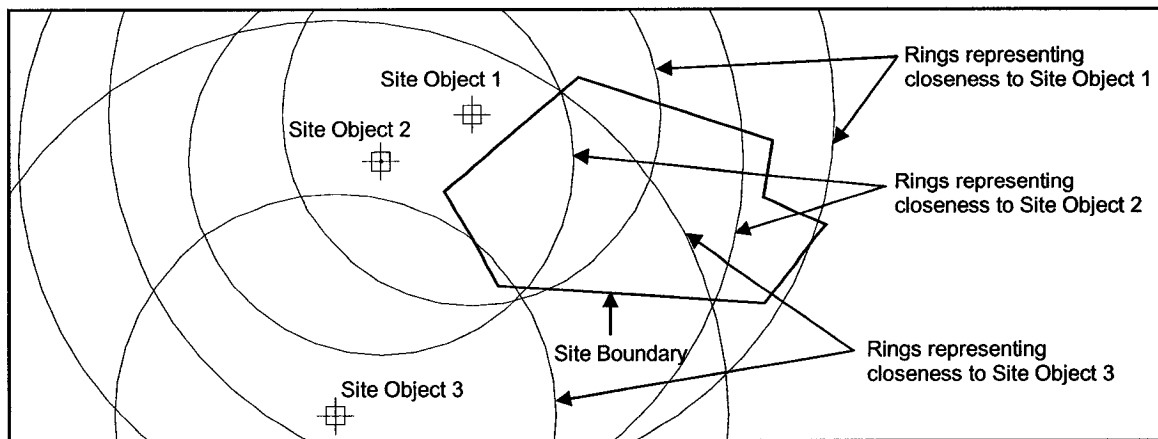


Figure 3.13. Rings representing the closeness to three objects divide the site area into smaller segments

Based on Equation 3.15 the utility score for each site segment is calculated and the segment with the highest score is identified. As well, other site segments are ranked according to their utility score to provide an analysis of the whole site area. In other

words, the planner is provided with a constraint satisfaction map, in which areas on site are ranked based on their overall constraint satisfaction score. This offers site planners a method of comparison for other locations on site to facilitate the decision-making process, in case alternative locations are needed.

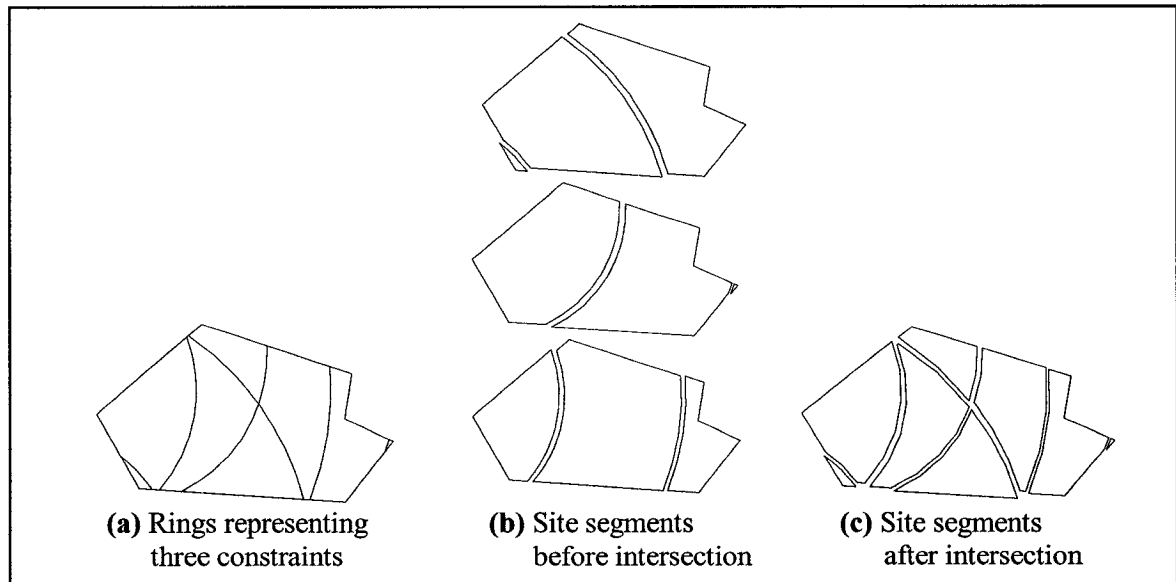


Figure 3.14. Dividing site area into smaller segments (a) Rings representing three constraints, (b) Site segments before intersection, (c) Site segments after intersection

3.6 SUMMARY

This chapter described problem representation for site layout modeling, focusing on two aspects: data structure and geometric space analysis. Object-based concepts were utilized to formalize data structure. Objects required to model a site layout project were introduced and their structure was described. Accordingly, the site layout problem is formulated as one of locating a set of *construction objects* while respecting the locations of *site objects* and satisfying the *constraint objects*. The data structure used in the

proposed methodology is based on the three tiers of objects referred to above. These objects, through their attributes and methods, support the functional requirements envisaged for site layout and affect the way site layout problem is represented. Objects are stored in their respective libraries from where they can be selected to setup projects. Based on this formalization, an open-architecture project setup was developed. The open architecture concept allows for creation of new objects when they do not exist in the libraries. As such it allows for direct contribution of experts to the system's libraries, which can later provide a decision support to the less-experienced planners. The adopted open architecture takes advantage of the formal structure defined for each type of objects. By having a formal structure, new objects can be easily created by assembling the right elements as defined by its structure. The open architecture defined here is applied to the project setup phase of the developed model, which is described in the next chapter. It makes the developed site layout model flexible and not limited, as in models developed by others, to a single project setting.

Further in this chapter, a geometry-based methodology for site space analysis was described. To facilitate user interaction in the layout process and aid in quick visualization, constraint satisfaction is represented graphically. For each constraint, the site is divided into regions representing different degrees of satisfaction for that constraint (CS). The shape and scheme of these site regions depend on the type of relation used to describe the constraint. Six types of relations along with their graphical representation were presented. Ascending and descending staircase and step functions are used to map these relations and measure the degree of constraint satisfaction. The advantage of the

graphical representation for constraints is that it alleviates the complexity of constraint satisfaction assessment from mathematical functions into a simple membership function, which can be easily recognized visually. As well, it supports interactive participation of users in layout process. Based on the developed constraint representation, a geometric space analysis method was developed to search for the optimum or near-optimum locations for construction objects on site. This methodology is inspired by human reasoning approach and is presented in a simple visual format to make the process easily comprehensible for planners. It is, as well, applied to the development of the site layout model described in the next chapter.

CHAPTER 4

PROPOSED MODEL

4.1 INTRODUCTION

Several descriptions of site layout tasks exist, each of which includes one or more activities of selection of facilities, sizing them, and locating them on site. The planner's formulation of site layout problem is essentially one of identifying required facilities and locating them on site while satisfying a set of constraints. Thus the task can be viewed as a rational operation of selecting a facility, identifying and selecting the space, and testing for constraint satisfaction. The search for the solution proceeds moving from one state to another through the problem space, starting from an initial empty site state, and ending at a state of a complete layout that satisfies the imposed constraints (Hamiani 1987). This chapter describes a site layout model developed to perform the identified tasks in a *flexible, comprehensible, and interactive* manner. The problem representation and modeling issues described in chapter three are applied in the development of the proposed model.

To enhance user comprehension, the model is designed to perform in a CAD environment. This facilitates visualization of physical objects and the layout, as well as the layout process and geometric reasoning of site analysis. The advantages of using CAD for construction applications to increase accuracy, improve communication of technical information, and increase field productivity have long been recognized

(Mahoney and Tatum 1994). A geometric model is an approximation of the physical objects. Different parties provide geometric information with various degrees of approximation. The use of a CAD tool allows utilizing the required precision in geometric representation and avoiding oversimplifications. Over simplifications in problem representation, such as limiting the shape of facilities to rectangles or their orientation to 0/90, is recognized to be one of the causes to render a site layout model impractical (Tam and Tong 2003). When space is limited, such misrepresentations can affect the feasibility of the layout by reducing the solution space to null. Using a CAD system removes the object representation limitations referred to in literature (Zouein 1996).

The importance of *flexibility* for a viable site layout model and its application through open architecture was discussed in the previous chapter. The developed model is designed to have an open architecture to allow creation of new objects. A relational database is employed to store the three defined tiers of objects. Database is an interrelated collection of different types of data stored in different tables. In a relational database, logical description of data is independent from its physical representation of data (Elmasri and Navathe 2000). This is referred to as data independence and ensures that modifications made to the physical representation do not affect the logical representation. Other motivation for the use of relational database are ease of use and flexibility with respect to operations, specially queries (Atzeni and De Antonellis 1993)

In knowledge-intensive tasks such as site layout, the *interaction* of user with the computer-based models is necessary. Humans are capable in grasping information and knowledge from experience, but they are poor at handling large quantities of data. As such, when analyzing problems, the limited symbolic processing power of the designer leads to decomposition of the problem into small manageable sub-problems (Hamiani and Popescu 1988). Computers on the other hand, are effective in manipulating large amounts of data, but need human knowledge as input to function and analyze problems; yet, their knowledge is limited to what has been granted to them. Thus, fully automated tools will not necessarily lead to acceptable solutions for knowledge-intensive problems (Zoueiri and Tommelein 1993). The developed model is designed to perform in an *interactive* manner to profit from the best of two worlds: knowledge and expertise of human and the ability of computers to handle and process large quantities of data.

Based on the representation described in the previous chapter, a CAD-based model was designed to assist site planners in developing site layouts. The proposed model, through its open architecture, calls for user interaction and allows for user intervention at different decision-making stages (Sadeghpour et al. 2002). This chapter explains the functionality and mechanism of the developed model and describes the interconnectivity among its modules.

4.2 MODEL ARCHITECTURE

Figure 4.1 shows the general structure of the model and the relation between its three basic components: Database, Project Module, and Layout Control Module. The two

modules are designed to perform and interact with the user in CAD environment. Each of the three tiers of objects is defined in the respective sub-module of the Project Module (i.e. Site Module, Construction Module, or Constraint Module), and stored in its respective library in model's database (i.e. "site library", "construction library", or "constraint library"). All objects defined for a given project are then passed to the Layout Control Module, which conducts the search to find the optimum location for each construction object on site and finally places them on the selected location on the layout. This chapter is allocated to detailed description of each module and its functionalities.

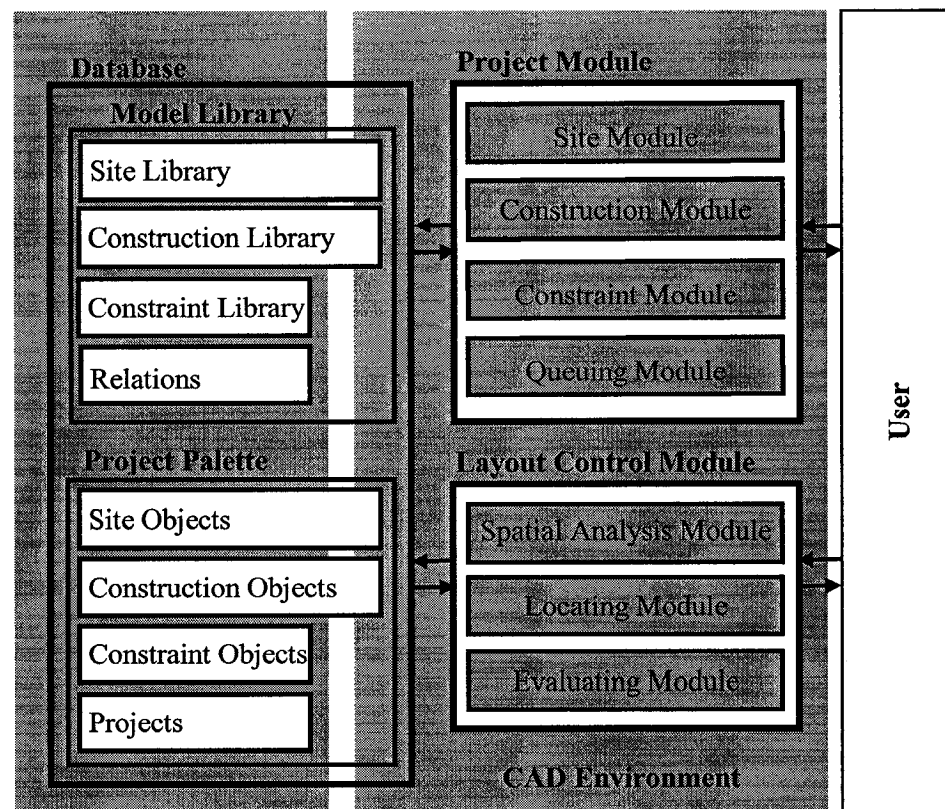


Figure 4.1. Model Architecture

4.3 DATABASE

A relational database is designed to facilitate the storage and retrieval of objects. Libraries of defined objects in the three categories of “site”, “construction”, and “constraint” are stored in the database (Figure 4.1). The design of site and construction objects includes properties -discussed in the previous chapter- and a graphical representation of the object generated in a CAD environment. CAD systems have a built-in database that keep record of graphical and geometrical properties of their objects such as area, perimeter, color, and line type. These properties can be accessed via object interface in the CAD system. The developed model takes advantage of this accessibility and merges this data with layout-related attributes stored in model’s database, external to the CAD system. As such, in terms of their storage location, the properties of the objects can be divided into *CAD* properties, including the graphical and geometrical properties, and *layout* properties. The link between the two databases is maintained through a unique identification property of objects. This property is automatically assigned by the CAD system to the graphical objects upon their creation. This unique ID is retrieved from the CAD system and recorded in the layout database to ensure connectivity between the two databases. In other words, the identification property is the overlap between the built-in database of the CAD system and model’s database to ensure connectivity between the two (Figure 4.2).

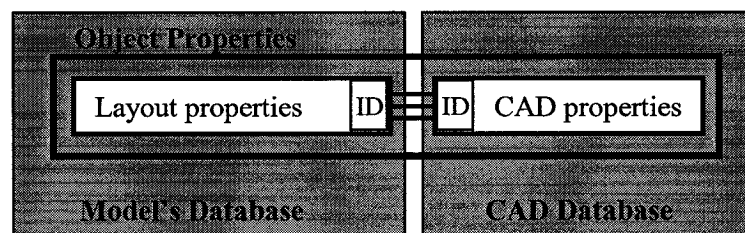


Figure 4.2. Connectivity between model’s database and CAD database through object ID

The libraries contain initial objects that can be selected to setup a site layout project. In addition, users can also add new objects in each of the respective libraries. This way the libraries get populated as the model is used. More importantly, this process addresses the needs and supports the planning strategies of each individual user. Selected objects will be added to the Project Palette (PP), which stores project-related data. As such, the database is divided into two functional sections (Figure 4.1). Model Library (ML) acts as an object gallery from which the objects can be viewed and selected; and Project Palette keeps record of the project-specific objects.

Figure 4.3 illustrates the process of selecting an object from database, and creating a new one and adding it to the Model Library. Each object is stored as a record in the database and its properties are the fields of database. When a site or construction object is selected from the library, first the object is added to the Project Palette; i.e. a record is created in the PP with a copy of the object's key field. Each graphical object has a "File Path" property in ML, which stores the location of the CAD file containing its geometry. This property is used to insert an instance of object's geometry into the layout. Once the geometry is generated, the CAD system assigns an ID to it. This ID is then added to PP to maintain the connectivity between CAD database and model's database (Project Palette and Model Library) as shown in Figure 4.3a. Creating a new object, on the other hand, starts with generation of its geometry in the CAD environment (Figure 4.3b). Its identification property (ID) is then added to PP to create a new record. This record, along with a set of *layout* properties, is added to ML to form an object in the library. The

geometry of the object is also saved in another CAD file, and its file path is added to ML for future use.

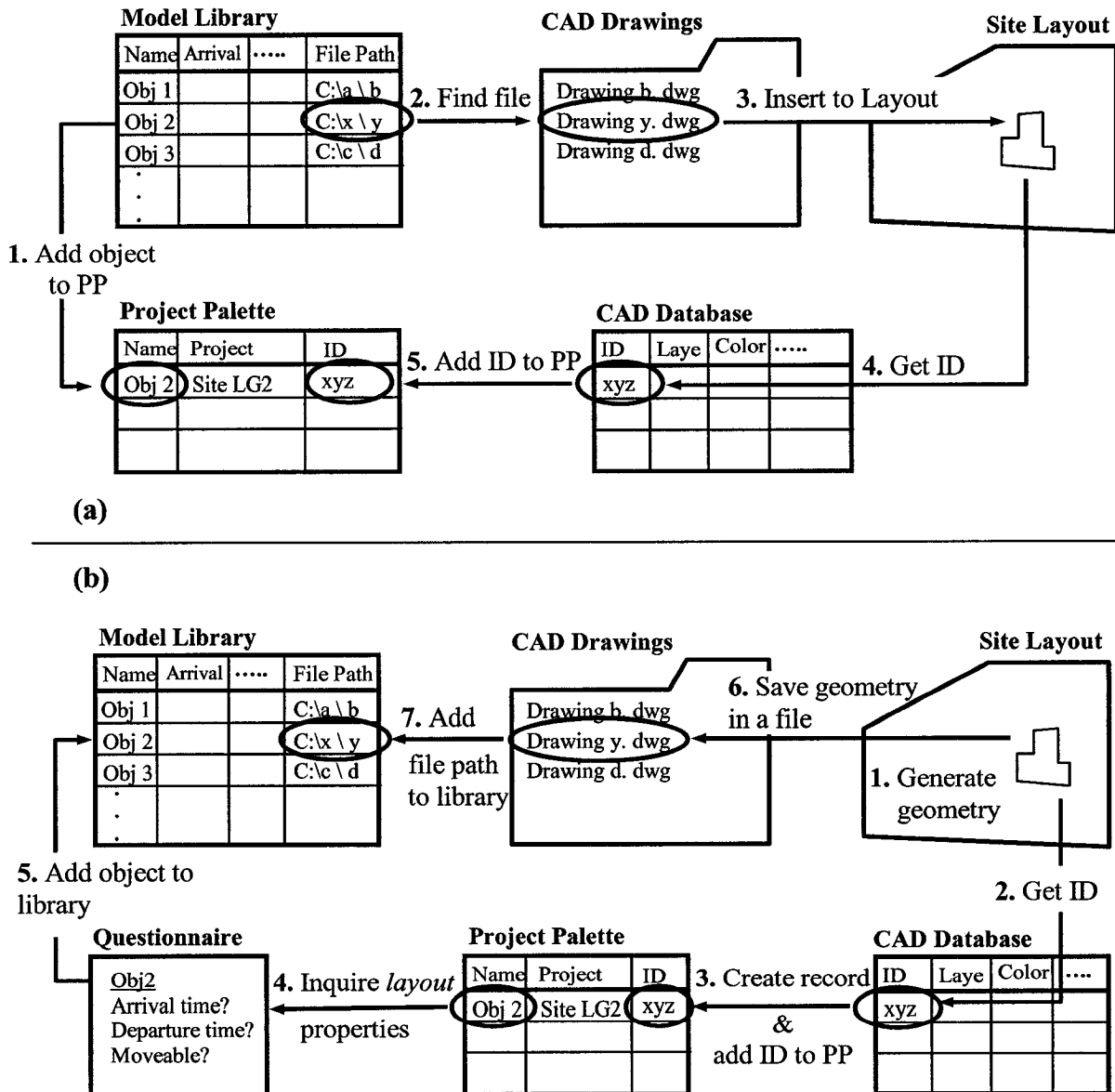


Figure 4.3. Defining objects (a) Selecting from library (b) Creating new objects

For constraint objects, selecting involves getting the required ones from the ML, and adding them to the “constraint palette” in PP (Figure 4.4). A record is created in “constraint palette” with a copy of the constraint object’s key field. To create a new constraint object it is sufficient to select its consisting elements from the database and add the new object first to ML and, from there, to PP (Figure 4.4).

Model Library

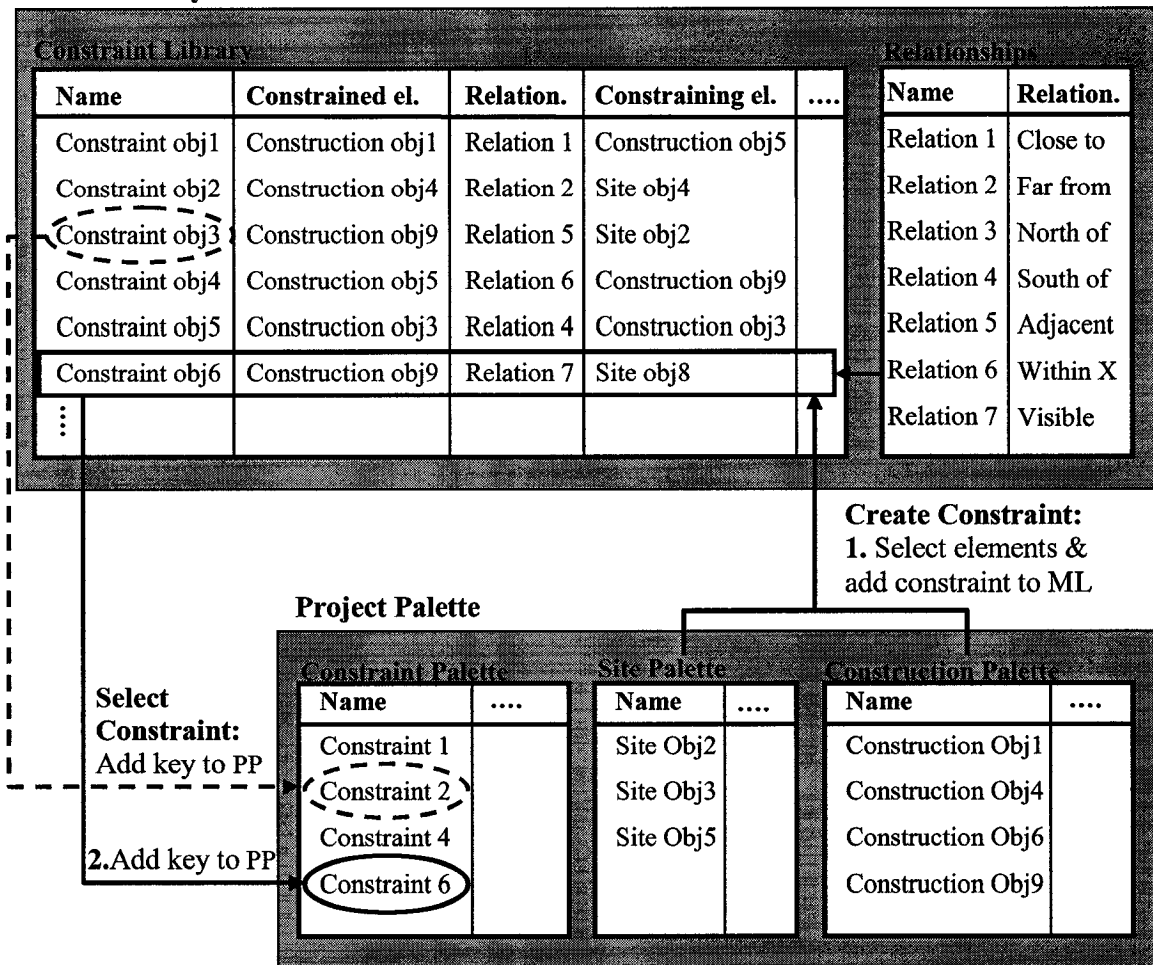


Figure 4.4. Defining constraining objects

Figure 4.5 depicts database entities and the relationships among them. The database consists of eight entities. In the Model Library “site library”, “construction library”, and “constraint library”, each contains respective tiers of objects. “Relations” stores the relationships that are used to create new constraint objects. “Site palette”, “construction palette”, and “constraint palette” in the Project Pallet, store records of selected objects for a project and “projects” keeps record of these projects.

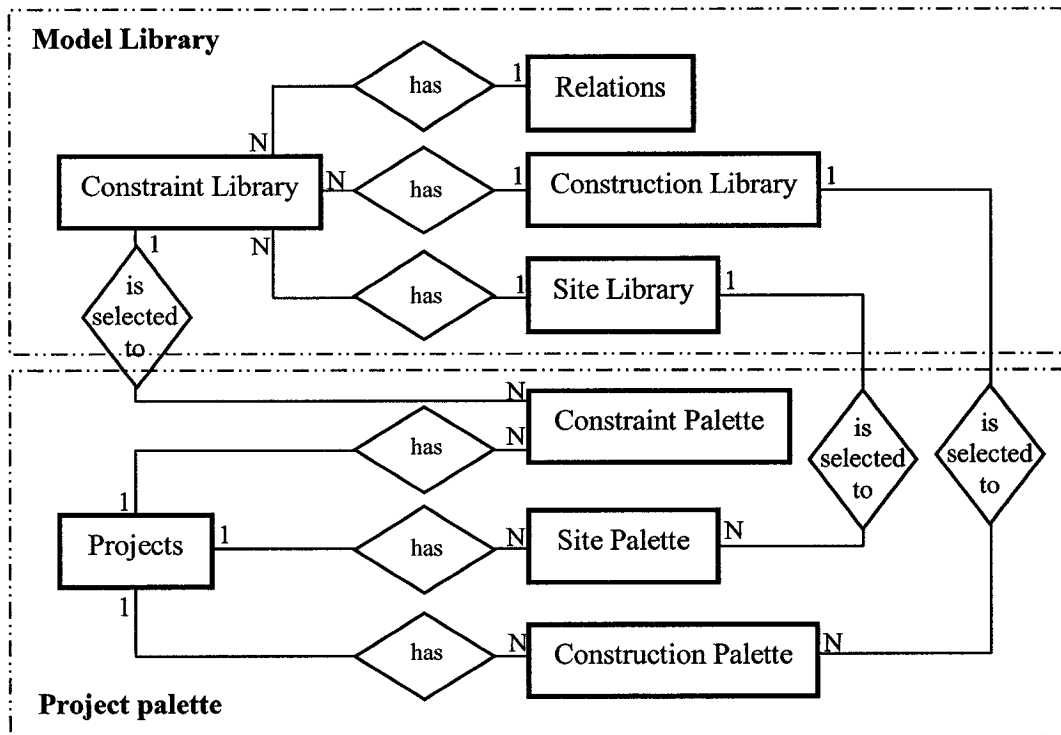


Figure 4.5. Entity relationship in the database

4.4 PROJECT MODULE

This module assists in defining the requirements of a site layout and setting up a new project. In this module the composing elements of the project are *defined* in each of the

respective sub-modules: Site Module, Construction Module, and Constraint Module, and are lined up in the Queuing Module in the order they are to be placed on site (Figure 4.6).

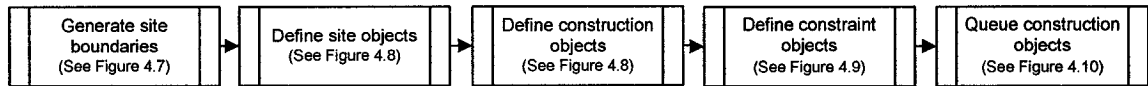


Figure 4.6. Order of activities in Project Module

Defining an object, refers to selection from a library, modification of existing objects, or creation of new ones. A record of *selected* object is sent to the Project Palette, representing the requirements for the project at hand. At this point the object can be modified if required, in which case it is added as a new object to the library. However, if the required object is not found in the libraries, the user has the option of *creating* a new one. Each time a new object is created, it is added to the corresponding library so that it is readily available for future use to supports the expansion and enrichment of the model's libraries and eliminates redefinition of objects. Since this expansion is project-based, it gradually customizes the model according to design needs and preferences of its users (Sadeghpour et al. 2003). The formal structure of each object explained in chapter 3 facilitates the creation of new objects in accordance with the open architecture design of the developed model.

4.4.1 Site Module

When *creating* an object, the model first prompts the user to define the geometry of that object. Depending on the case, this can be carried out using: 1) the readily available geometry of that object; 2) model-assisted drawing environment; or 3) the CAD system (Figure 4.7). If the geometry of the object is readily available in a drawing file, it is

directly inserted into the layout. Otherwise the geometry has to be generated. The coordinates of the corners of the object's geometry can be given as text input and the model generates the geometry. As an alternative, the model also accepts length of sides of the object and the angles between them. If it is selected to employ the CAD system is, the model temporarily passes the control to the CAD system and allows for drawing directly on the work area. The three aforementioned methods provide a flexible way to generate irregular shapes as required by project, and hence the shape of the objects is not limited to rectangular or rigid formats. Once the geometry of an object is generated, the model then prompts the user to input its *layout* properties. The *layout* data acquisition is conveyed in a text format questionnaire, inquiring for values for layout properties of site object listed in table 3.1. These properties are stored in a record in the “site library” and have a link to the “site palette” (Figure 4.3).

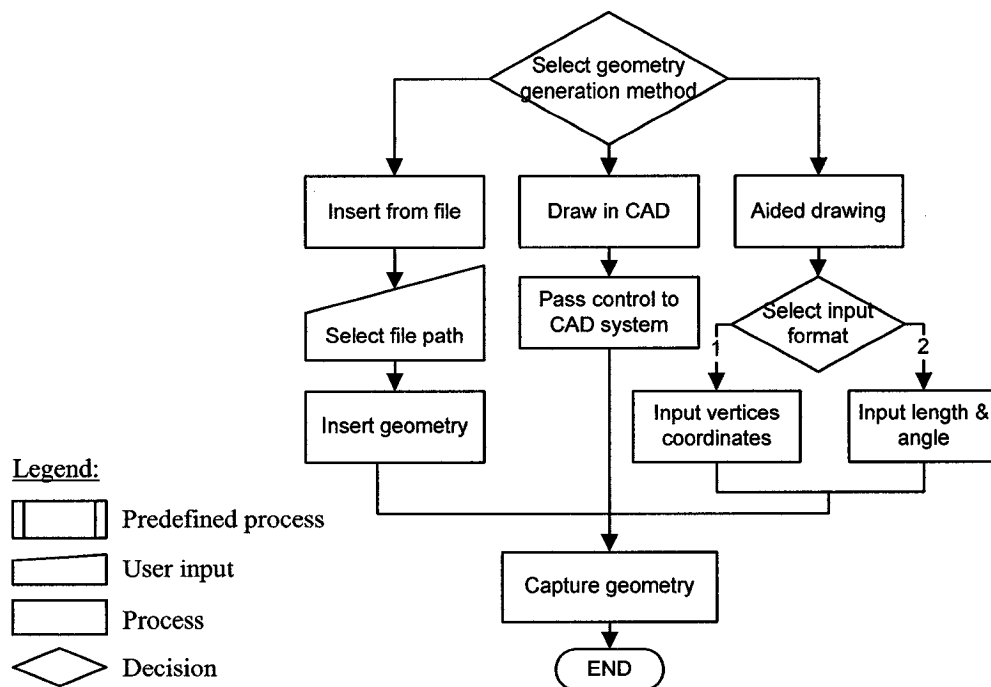


Figure 4.7. Generating geometry of objects

When a site object is *selected* from the library, an instance of its geometry is generated on the work area. The location of the CAD file is provided by the “file path” property of the site objects, from where it can be inserted into the layout. Since the location of site objects is predetermined, they get located on their final position on the site. As mentioned, at this point a unique ID is assigned to it from the CAD system, which is recorded in the “site palette” to ensure connectivity with the CAD database. As such, for each site object selected for the project at hand, the layout properties are coupled with the geometrical and graphical ones read from the built-in database of the CAD system. There is a two-way link between the record and its corresponding graphical object that appears on the layout. By knowing the ID of an object, its geometrical and graphical properties such as area, perimeter, or centroid can be obtained from the built-in database of the CAD system. Conversely, through the ID property, the layout properties of an object can be retrieved, modified, and saved from within the CAD system.

4.4.2 Construction Module

Construction module is in charge of defining construction objects required for a project. Construction objects are defined in the same manner as the site objects (Figure 4.3), with the exception of their insertion point on the layout. Unlike site objects, the location of construction objects is not known at the project setup phase and hence their geometry is inserted on the side of the layout, outside the site area, waiting to be assigned a location in the Layout Module. Figure 4.8 illustrates the functionality of construction and site sub-modules.

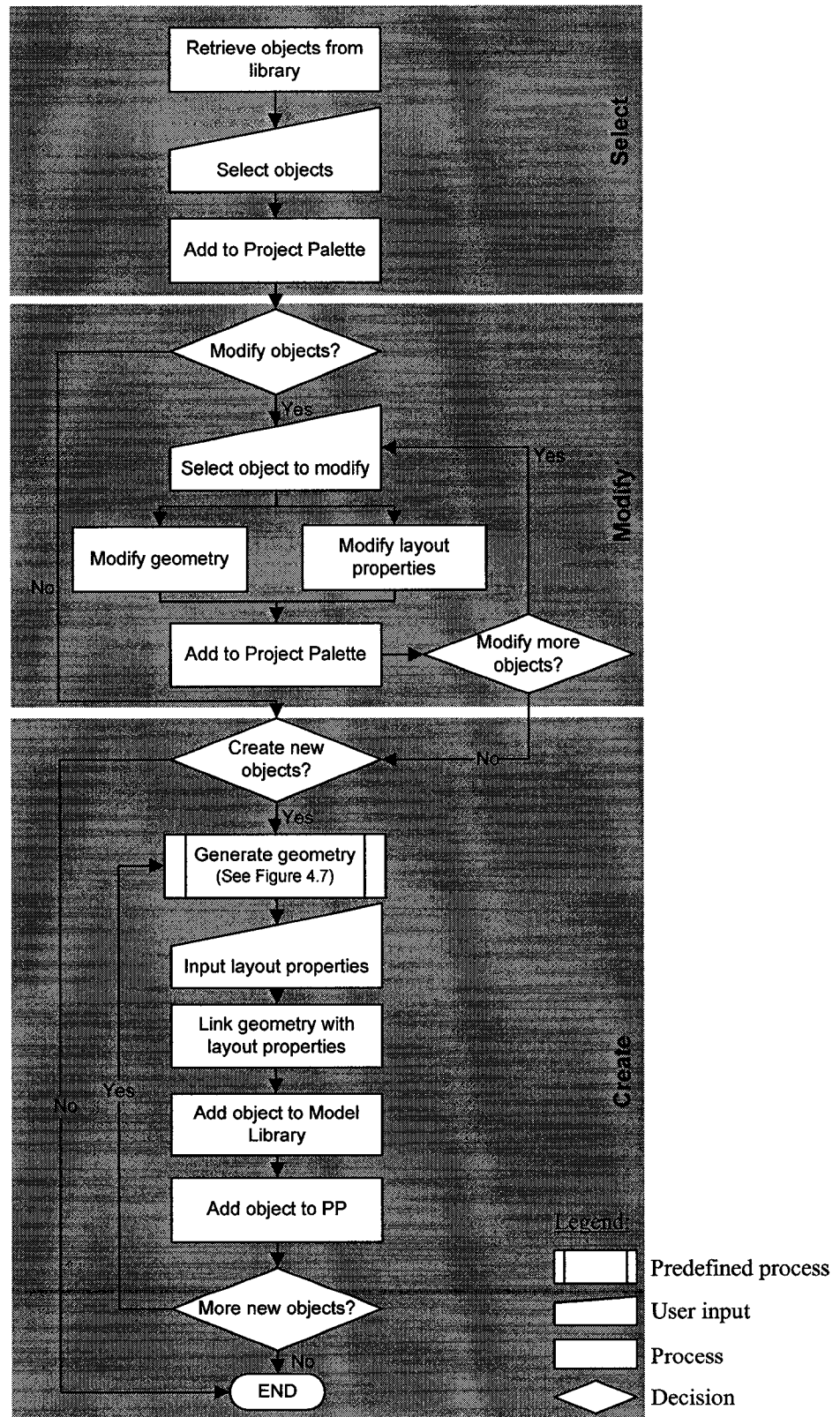


Figure 4.8. Dataflow in site and construction sub-modules

4.4.3 Constraint Module

The constraint module outlines the interconnectivity among site and construction objects using a set of constraint objects. These objects are rules and preferences designed to find a near-optimum location for each construction object. The structure of a constraint object, consisting of *constraining element*, *constraint element*, and *relationship* was described earlier in chapter 3. A set of constraint objects is *defined* for each construction object that has been selected and added into the “construction palette”. As such, the construction object in hand forms the *constrained element* in that constraint. The order of activities carried by “constraint module” is summarized in Figure 4.9. When *selecting* constraints to assign to a construction object, the model retrieves a list from the “constraint library”. This list includes constraint objects whose *constrained element* is the construction object in hand. Before adding a selected constraint object to the “constraint palette”, the model verifies if its *constraining element* (i.e. a site or construction object) is already part of the Project Palette. If it is not, the model requires the *constraining element* to be added to the Project Palette. This ensures that there are no discrepancies in the list of objects in the PP.

For *creating* a new constraint, it suffices to select its composing elements from Model Library. The construction object, for which the constraint is being defined, forms the *constrained element*. The *relationship* is selected from list of “relations” table, located in ML. The *constraining element* can be chosen from the list of construction or site objects in the “site” or “construction palette”, respectively (Sadeghpour et al. 2003).

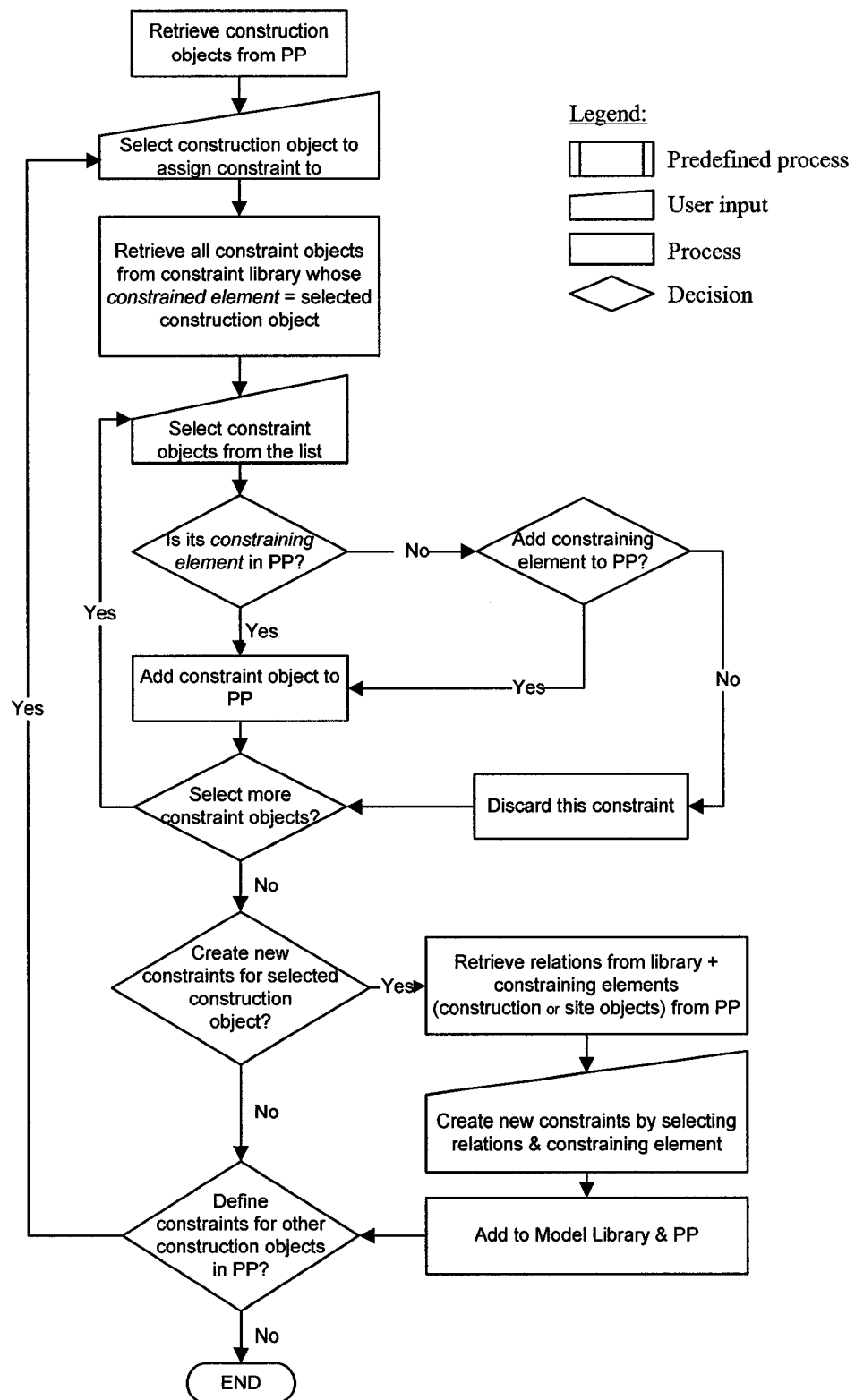


Figure 4.9. Dataflow in the constraint sub-module

4.4.4 Queuing Module

The layout method used in the developed model is a construction method, as opposed to improvement method. This means the model adds one object at a time to the layout, considering the updated status of the site before adding each object. In this method the order in which the construction objects are located on site affects the final layout, since the location of one impacts the location of subsequent entering objects. In deciding the order in which objects enter the layout, Hamiani (1987) suggests distinction based on project type. In projects that quality assurance is required (e.g. power plants), material-oriented facilities such as warehouses are critical and thus are given priority. Where quality assurance is not a critical factor, labor-oriented facilities (e.g. offices, change house) are given a higher priority. Tam et al. (2002) have applied fuzzy logic to assess the priority order of objects. This method is based on pair-wise comparisons between objects, but does not provide a method to compare all the objects at one instance. To enable comparison of all objects together, others have suggested random selection of objects (Zouein 1996) or to order them based on a single attribute such as “highest closeness relationship” (Moore 1971), “area requirement”, “duration on site”, or “cost of relocation” (Zouein 1996), without articulating a formal structure for the collective application of these heuristics. The latter research points out the need for a dynamic and real-time process for selection of heuristics for queuing objects based on the uniqueness of the project and judgment of site planner. Thus, it is important to ensure that the planner is able to queue objects based on experience, intuition, and due considerations of applicable site constraints. In an effort to achieve such dynamic and user interactive process in the proposed model, two methods are developed for queuing objects. In one,

named *independent* method, objects are queued based on their properties, and do not have interdependencies. In the other, called *dependent* method, the positions that objects get on the queue are inter-dependent and cannot be determined without knowing the preceding objects. The mechanism of the independent and dependent methods is described in this section.

Independent Queuing Method: Assignment of priorities in the independent method is established based on a multi-attributed queuing score (Q). This score accounts for a combination of weighted heuristics such as those referred to earlier (i.e. “area requirement”, “duration on site”, “cost of relocation”), as well as other factors such as “number of constraints”, “total weight of constraints”, “time of arrival”, and “number of *constrained elements*”. The latter refers to the frequency of the object being selected as a *constraining element* in a constraint object. The planner selects applicable attributes to be used in the queuing process of each project and assigns a relative weight to each, indicating its importance compared to others.

The multi-attributed queuing score (Q) for each construction object is calculated using Equation (4.1).

$$Q = \sum_{i=1}^n \frac{C_i}{C_{\text{Max}}} \cdot W_{C_i} \quad 4.1$$

where C_i is the value associated with the i th attribute, C_{Max} is the maximum value of that attribute for all objects, and W_{C_i} is the weight assigned to this attribute. Accordingly,

objects are queued in descending order of their queuing score. The planner can choose to proceed with that queue or modify the order of its objects (Figure 4.10). As well, different combinations of the attributes, leading to different orders of objects, can be invoked to study the impact on the final layout. Once the queue of construction objects is determined, it is passed to the Layout Control Module to place the objects on site.

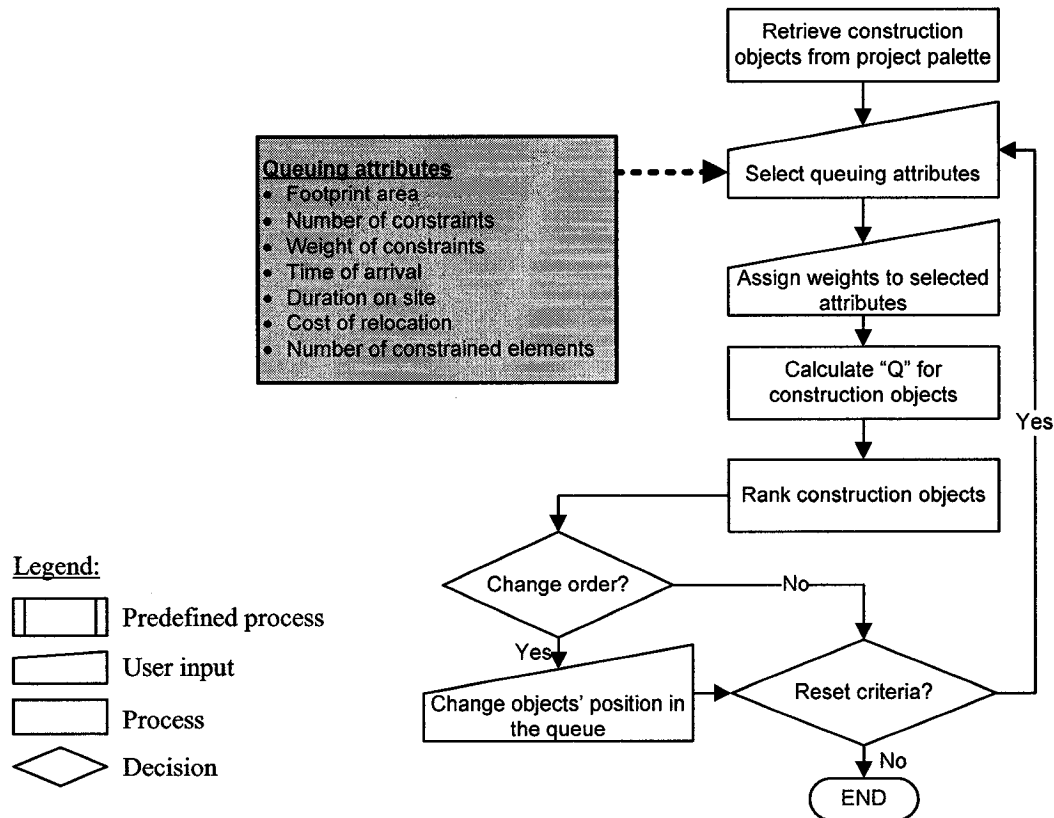


Figure 4.10. Independent queuing of objects in the Queuing Module

Dependent Queuing Method: Unlike the previous method, in dependent queuing method the position of the construction objects in queue cannot be determined at one instance. Rather, the assignment of an object to a position is dependent on preceding objects in the queue and, thus, cannot be decided before its predecessors. In fact, the

assignment of an object to a position, affects the rest of objects in the queue, hence the name dependent. The assignment is determined for one object at a time, considering the updated status of the queue. Two methods for dependent queuing of construction objects are developed using one of the following rules:

1. Most number of located *constraining elements* (Max LCE)
2. Least number of unlocated *constraining elements* (Min UCE)

Both rules consider the constraining elements of the constraints assigned to the construction objects. Constraining elements are important for locating priorities since in fact their location on layout guides the location of the constrained element. The Max LCE rule gives priority to the construction object that most number of its constraining elements has already been located on site. With more number of constraining elements on site, it is easier to delimit the neighborhood of a location for the construction object that satisfies its constraints. The object with the highest priority can itself be *constraining element* for one or more of the remaining objects. In that case, as this object is assigned a position in the queue, it affects the number of LCE (located constraining elements) for the remaining of objects. Therefore the priority of remaining objects should be revised with the assignment of every object to a position in the queue. Similarly, the Min UCE rule gives priority to the object that has the least number of constraining elements that are not located on the layout yet. The logic here is similar to the previous; when searching for an optimum location for a construction object with less number of unlocated constraining elements, less of its locating constraints are not considered in the procedure. Hence, better overall constraint satisfaction is expected.

The aforementioned rules seem similar. Indeed, for *one* construction object, more LCE means less UCE. However, these rules are to compare the LCE and UCE between *a number of* construction objects. Since the total number of constraints for all the construction objects is not the same, the two rules do not necessarily result in the same queue of objects. Consider constraint objects C_1 through C_{19} formed of construction objects A through F, site objects S_1 , S_2 , and S_3 , and relationship x , as presented in Table 4.1.

Table 4.1. Composing elements for the constraint objects

Constraint	<i>Constrained element</i> (Construction object)	<i>Constraining element</i> (Construction/Site object)	Constraint object
C1	A	C	A x C
C2	A	D	A x D
C3	A	S1	A x S1
C4	A	S2	A x S2
C5	B	A	B x A
C6	B	E	B x E
C7	B	C	B x C
C8	C	D	C x D
C9	D	S1	D x S1
C10	D	S2	D x S2
C11	D	E	D x E
C12	E	S1	E x S1
C13	E	S2	E x S2
C14	E	S2	E x S3
C15	F	A	F x D
C16	F	B	F x C
C17	F	C	F x B
C18	F	D	F x A
C19	F	E	F x E

Table 4.2 contains the number of LCE for the construction objects A-F, in competition to get each position in the queue. To determine the first object entering the layout using the LCE method, only *constraining* site objects are counted since they are the only

constraining elements located on site at the commencement of the site layout process. Object E gets the highest number of three site objects amongst its constraining elements; hence it gets the highest priority in the queue (Table 4.2). Locating object E adds one more object to the LCE for objects B, D, and F (see Table 4.1). Therefore in competition for the second position in the queue, object D gets the priority with three LCE. Carrying the same logic, the order of objects in the queue using the Max LCE is E-D-A-F-B-C (see Table 4.2).

Table 4.2. Queuing construction objects using Max LCE rule

Construction object	Position in Queue					
	1 st	2 nd	3 rd	4 th	5 th	6 th
A	2	2	3	---	---	---
B	0	1	1	2	2	---
C	0	0	1	1	1	1
D	2	3	---	---	---	---
E	3	---	---	---	---	---
F	0	1	2	3	---	---
Queue	E	D	A	F	B	C

Table 4.3. Queuing construction objects using Min UCE rule

Construction object	Position in Queue					
	1 st	2 nd	3 rd	4 th	5 th	6 th
A	2	2	1	0	---	---
B	3	2	2	1	0	---
C	1	1	0	---	---	---
D	1	0	---	---	---	---
E	0	---	---	---	---	---
F	5	4	3	2	1	0
Queue	E	D	C	A	B	F

This order changes when the Min UCE rule is applied. Here, counting of UCE starts with the *constraining* construction objects, since they are not located at this point. Hence E gets the highest position in the queue with the lowest UCE of zero (Table 4.3). With E getting the first position in the queue, the UCE for objects B, D, and F decreases (see

Table 4.1) and D with no UCE gets the second position in the queue. Consequently, with D in the second position, the UCE for objects A, C, F decreases, and object C gets the third position. Applying the Min UCE rule to the rest of objects yields the order of objects in the queue as E-D-C-A-B-F (see Table 4.3).

Clearly, there are cases where there is a tie in the competition for getting a position in the queue. For example, consider removing the constraint object C_3 ($A \times S_1$) from the previous list of constraint objects (see Table 4.1). Tables 4.4 and 4.5 show the tie between objects A and F on taking the third position in the queue when the Max LCE method is applied. Tables 4.4 and 4.5 respectively illustrate the scenario if objects A or F are selected.

Table 4.4. Conflict on Max LCE. Case
1- Select A

Construction object	Position in Queue					
	1 st	2 nd	3 rd	4 th	5 th	6 th
A	1	1	2	---	---	---
B	0	1	1	2	2	---
C	0	0	1	1	1	1
D	2	3	---	---	---	---
E	3	---	---	---	---	---
F	0	1	2	3	---	---
Queue	E	D	A	F	B	C

Table 4.5. Conflict on Max LCE. Case
2- Select F

Construction object	Position in Queue					
	1 st	2 nd	3 rd	4 th	5 th	6 th
A	1	1	2	2	---	---
B	0	1	1	1	2	---
C	0	0	1	1	1	1
D	2	3	---	---	---	---
E	3	---	---	---	---	---
F	0	1	2	---	---	---
Queue	E	D	F	A	B	C

It is interesting to note that removing constraint object A x S1 from the list of constraints does not change the order of objects when applying Min UCE rule. In other words removing C₃ does not change the scenario and queuing order presented in Table 4.3. This is due to the fact that the *constraining element* of this constraint is a site object (i.e. C₃ is a C-S constraint). Since the location of all site objects is known from the beginning of a project, they are not considered when counting the number of unlocated constraining elements (UCE). Hence, any changes in the number of C-S constraints will not affect the order of objects when applying Min UCE. The dependent method presented here can be used as an alternative to the previous method for queuing construction objects. As well, when objects have competition for the same position in the queue, either of the methods can be used as a tie-breaking technique for the other.

4.5 LAYOUT CONTROL MODULE

Once the project is configured in the Project Module, the queue of objects is sent to the Layout Control Module. This module deals only with the objects in the Project Palette, which represents the specific requirements of the project at hand. Three major tasks are performed in this module: analyzing site space to find the optimum or near the optimum location for each construction object in the queue; placing the construction objects on site; and evaluating the layout against an objective function. Each of these tasks is performed by *spatial analysis*, *locating*, and *evaluating* sub-modules, respectively.

4.5.1 Spatial Analysis Module

The spatial analysis sub-module scans the available site area to find the best location based on the knowledge and information provided by the Project Palette. The process of finding the location is performed through a geometric reasoning search described in chapter 3. When locating a construction object, the model retrieves all the constraint objects assigned to it. This translates to all the constraint objects in PP whose constraining element is the construction object at hand. Depending on the relationship selected for these constraints, the site is divided into segments. The discretization method for each relationship is described in section 3.5.1. A utility score for each site segment is measured as sum of satisfaction scores for each constraint, using Equation 3.15.

The site segments obtained by the geometric representation of constraints are irregular in size and shape. The utility score for site segments is calculated at their respective centroid, which form an irregular network of grid points. The size of the solution segment found by the model depends on the selected offsetting width in linear relationships and indicates the precision of the answer. The smaller the offset width is chosen, the smaller the size of the grid is in general, and consequently the more precise is the location found for the construction object. If higher precision is required, the model allows for refining the analysis within the solution segment. The grid refining process can take place in several iterations, zooming down on the near-optimum location, until the precision of the location, and in other words the size of the solution segment, is deemed satisfactory. The planner can accept the location found by the model, or veto the model's solution and select another location based on the analysis and results provided by the model. The grids

and geometries resulted from each analysis iteration are stored in a separate layer and can be viewed separately or simultaneously to assist the planner in deciding the final location of the object at hand. If a location other than those ranked by the model (i.e. centroid of segments) is selected, the model can calculate the utility score of that location for comparison purposes.

As an alternative to the discretization method, the model can analyze the site space utilizing a generic rectilinear grid (Sadeghpour et al. 2004b). The size of the grid can be selected according to the required precision. Elbeltagi and Hegazy (2003) suggest that the grid size should follow the Greater Common Divisor (GCD) of all facilities to be located on site; or to be equal to the largest object (Hegazy and Elbeltagi 2000). In these research works, the shape and/or size of objects are dependent on those of the grid. In the proposed model, however, the grid is utilized as a means for site analysis and does not impose limitations on the size and shape that construction objects can take. As a result a different grid size can be selected when analyzing the site for each constraint object. Similar to the previous method, in order to speed up computations, the developed system can perform space analysis in several iterations. The search can start with a larger grid size and gradually zoom down in search for a near optimum solution by building progressively finer search grid in each iteration. The refining process can continue until the targeted precision is achieved. As a rule of thumb, when locating a construction object, the grid size can be selected as the size of the object in hand.

Another flexibility step counted for in the model is the distance measurement method. Besides the Euclidean distance measurements used to present constraint satisfaction scheme, the model is able to employ rectilinear distances if it is deemed to be more suitable for the site and project conditions. However, it should be noted that both measurements are simplifications of the actual path taken between two objects. In practice, due to the various obstacles, as well as the use of established routes on construction sites, the actual path taken between two objects is different from either of the measures and is out of the scope of this research.

The graphical features of the CAD system are utilized to make the layout process more comprehensible. To facilitate recognition, a color scheme is used to represent the level of constraint satisfaction for each site segment. Each segment (or grid point in the rectilinear grid approach) is assigned a color according to its utility score. As such, besides identifying the fittest location, the model provides a visualization of the analysis of the whole site area. As a result, if for any reason the user does not approve of the identified segment, a method of comparison for other locations on site is provided to support the decision-making process for site planners (Sadeghpour et al. 2004a). When the location of an object is finalized and the model starts locating next object, the layer(s) containing the geometries and graphics related to the site analysis for the previous object gets hidden and a new layer for the object in hand is created. These layers can be reviewed later for further studies and creation of alternative layouts.

4.5.2 Locating Module

Once the final location of a construction object is decided, the locating module proceeds with laying the object on the identified location. By default, the model matches the centroid of the construction object in hand with that of the segment identified for the object's location. This location can then be fine-tuned as required using the following three features. 1) Corner-to-corner adjacency allows matching one of the corners of the *bounding box* of object in hand with one of that of an existing object on site. Hence, considering four corners of each bounding box, sixteen positions can be acquired (Figure 4.11a). 2) Vertical and horizontal alignments adjust the alignment of sides, or midpoint of sides, of the bounding box of the construction object at hand with those of an existing object on site (Figure 4.11b). 3) Rotation option revolves the construction object round its centroid by an indicated angle.

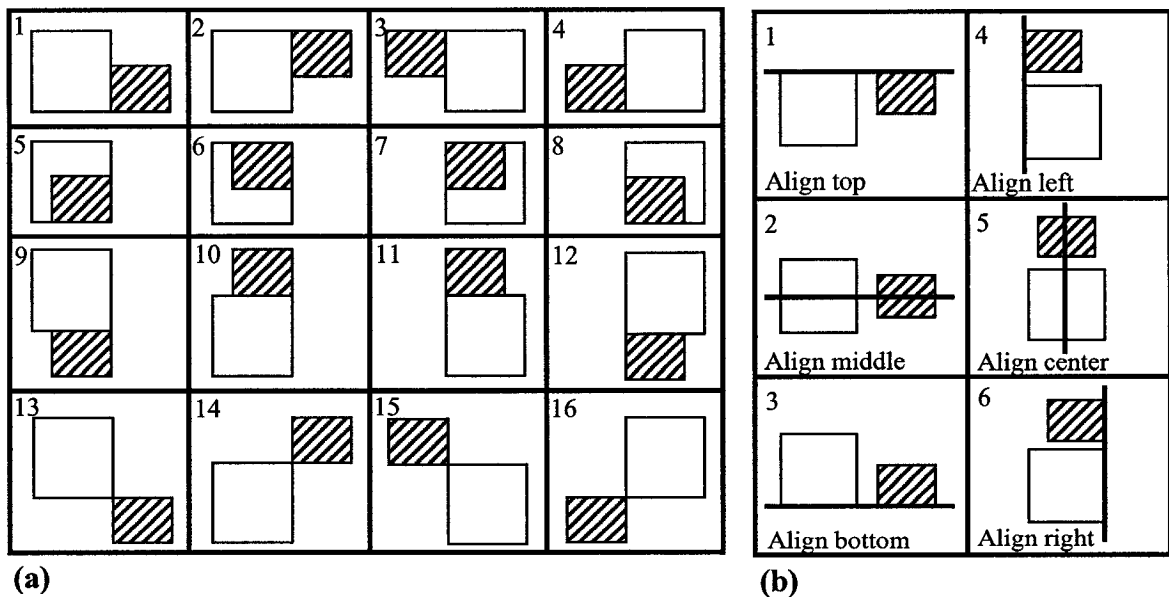


Figure 4.11. Alignment of objects (a) Corner-to-corner adjacencies (b) Vertical and horizontal alignments

If required, the Locating Module allows for modification of object's geometry. Some construction objects such as parking spaces or material storage area are more flexible in shape and can be reshaped according to the location they are situated in, provided the same area is maintained. The model verifies the flexibility property of the object to determine if the geometry can be modified. If the object is *flexible* or *sizeable* the modification is allowed, but the shape of *rigid* objects cannot be modified. Further, to satisfy the non-overlap condition, the locating module verifies whether the object overlaps with an existing object on site. The model allows the overlap between objects if the flexibility property of at least one of them is set to *flexible* or *sizeable*. In that case, as mentioned earlier, the planner still has the choice of re-shaping to avoid overlap condition.

Once the position of a construction object on site is finalized, the locating module updates the status of site area. It deducts the footprint area of the newly located construction object from the available site area. For the next construction object in the queue, only the updated available site area is analyzed. This deduction prevents the allocation of committed areas to succeeding construction objects.

4.5.3 Evaluating Module

Once all the construction objects are located on the layout, the total utility score of the layout can be calculated as:

$$U_t = \sum_{i=1}^n \sum_{j=1}^m S_{ij} \cdot W_{ij} \quad 4.2$$

Where n indicates number of construction objects, m is the total number of site and construction objects, S_{ij} is the satisfaction score of the constraint defined between the i th and j th object; and W_{ij} is the weight of this constraint. Clearly, the quality of site layout is not easy to measure. Many factors are involved in a layout process and not all of them are quantifiable. The total utility score defined here is a measure of satisfaction of the constraints defined by the planner for a project. This function is useful in comparing the fitness of different layouts. Such layouts can be generated by changing the order of construction objects in the queue and reanalyzing site space according to the new order. As well, once all the construction objects are located on site, different layouts can be created by altering the location of objects on layout. The total utility score for each layout is measured to identify the fittest among them. Figure 4.12 illustrates a summary of the main activities performed in each module of the developed model.

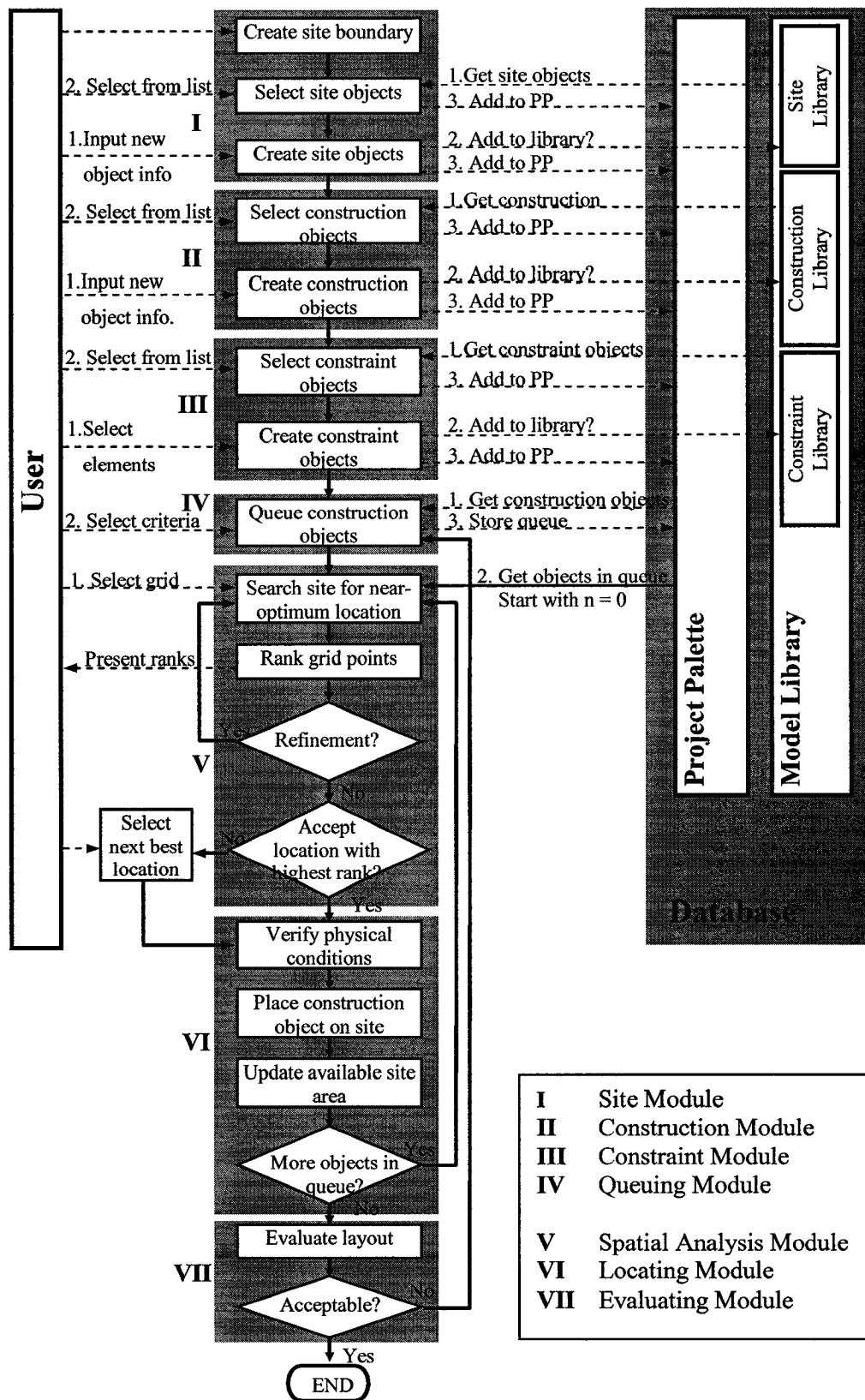


Figure 4.12. Dataflow in the developed model

4.6 SUMMARY

This chapter presented a code-independent description of a CAD-based model for site layout planning along with its general architecture. The basic components of the system and the inter-connectivity among them were described. The model consists of three components: database, Project Module, and Layout Control Module. The Project Module assists in defining the requirements of a project by defining three tiers of objects. Also in this module the connectivity between the objects are defined, i.e. the *constraints objects* between *construction objects* and *site objects* are defined. Two queuing methods were proposed to aid in determining the priority order of objects for entering the layout. Once the requirements and settings of the project are defined, they are passed to the Layout Control Module to analyze site space and find optimum or near-optimum locations for construction objects. Project Palette, which is the project-specific section of database, keeps track of the objects selected and created for the project at hand. The developed structure for three tiers of objects facilitates the creation of new ones in accordance with the open architecture of the model. The open architecture allows for the incorporation of user-defined objects based on the requirements of the project, if the model does not readily offer them. Newly created objects are added to the Model Library, which acts as an object gallery in the database.

The use of CAD, database, and open architecture in the design of model grants it a number of interesting features: 1) It provides a flexible support of a wide range of objects for site planner. This permits the set up of different construction projects via selecting objects from three libraries. 2) The system has a built-in feedback to support the

development of new objects and/or updating existing ones. This allows for the gradual expansion and enrichment of the system's database and the supporting libraries. 3) Site space analysis is performed geometrically. This feature facilitates easy visualization of site planning process and provides a range of ranked near best solutions to encourage user participation in the layout process. 4) The model is designed to be highly interactive and allows site planners to decide and interfere on every aspect of process, such as search criteria, based on their knowledge and expertise. It also allows for experimenting with different rules and comparing the final layout results. The developed model could essentially be viewed as a space planning tool that is compliant with common industry practice and allows to account for a set of constraints related to productivity, safety, and security.

CHAPTER 5

IMPLEMENTATION

5.1 IMPLEMENTATION ENVIRONMENT: AUTOCAD-VBA-ACCESS INTEGRATION

A prototype of the model described in the previous chapter was implemented as proof of concept. This prototype is called CASL ['kæsəl], which stands for Computer-Aided Site Layout. The prototype was implemented using Visual Basic for Applications (VBA) in AutoCAD® environment, and employing Microsoft Access® to develop the model's database. AutoCAD and Microsoft Access are two of the most commonly used tools in AEC industry. VBA provides these tools with a flexible programming interface, and renders them suitable for fast prototyping. VBA is an object-oriented programming environment with development capabilities similar to those of Visual Basic (VB). VBA runs in the same process space as AutoCAD, providing seamless integration and fast execution of the program. It also provides integration with other VBA-enabled applications such as MS Access. This enables AutoCAD to be an automation controller for such applications, while being able to use their respective object libraries (Autodesk 2001). As such, VBA provides a seamless link between the main components of the model, supported by a powerful graphical user interface (GUI).

VBA is granted access to AutoCAD objects through ActiveX. AutoCAD ActiveX technology provides a mechanism to manipulate AutoCAD programmatically from within or outside AutoCAD by exposing AutoCAD objects to other environments. Once

exposed, these objects can be accessed by other programming languages and applications such as Microsoft Access[®] VBA (Figure 5.1). In other words, Active X provides a platform for AutoCAD drawings to be accessed by other programming environments and facilitates data sharing with other applications. As such activeX enables cross-application macro programming, a capability that does not exist in other programming languages of AutoCAD. Thus, through ActiveX, features of different applications can be combined into a single application. The exposed objects are called AutoCAD ActiveX objects. An object is the main building block of any ActiveX application, representing a precise part of AutoCAD. Different types of objects exists in the AutoCAD ActiveX interface such as graphical objects, style setting objects, organizational objects, display objects, and even the drawing itself (Autodesk 2001).

Through the exposed objects, their methods and properties can be accessed. Methods perform an action on an object while properties set or return information about the state of an object. Often the terms *method* and *function*, as well as *property* and *attribute*, are used interchangeably. However, in this chapter these terms are used distinctively. Here, the term *method* is used for methods/functions available in AutoCAD and VBA programming, whereas *function* is used to address those developed for the implementation of CASL. As well, the word *property* is used for those of AutoCAD objects, referred to as “CAD properties” in the previous chapters; and *attributes* refer to those stored in the database, addressed to as “layout properties” previously.

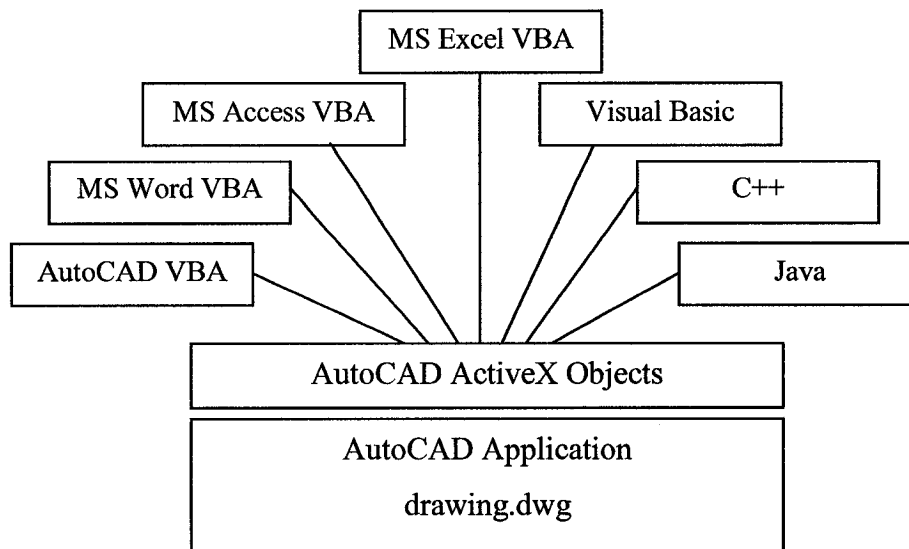


Figure 5.1. AutoCAD ActiveX technology (Autodesk 2001)

Three fundamental elements define ActiveX-VBA programming in AutoCAD; 1) AutoCAD, which has a rich set of objects encapsulating AutoCAD entities, data, and commands. 2) AutoCAD ActiveX Automation interface, which establishes messages with AutoCAD objects. This interface is described through AutoCAD object model in the next section. 3) VBA programming environment, which has its own set of objects, keywords, and constants that provides program flow, control, debugging, and execution. VBA sends messages to AutoCAD through the AutoCAD ActiveX Automation interface. AutoCAD VBA permits the VBA environment to run simultaneously with AutoCAD and provides programmatic control of AutoCAD through the ActiveX Automation interface. This coupling of AutoCAD, ActiveX Automation, and VBA provides a powerful interface for manipulating AutoCAD objects, as well as sending data to or retrieving data from other applications. VBA can interact with Access through Structured Query Language (SQL) commands. As such, in addition to manipulating the data through read-

write functions (load, save, edit, and update), CASL is able to perform queries on Access tables from within AutoCAD.

5.2 IMPLEMENTATION ISSUES

5.2.1 AutoCAD Object Model

A partial AutoCAD object model illustrating the hierarchy of the objects used in the implementation of CASL is shown in Figure 5.2. The complete object model of AutoCAD is presented in Appendix II. AutoCAD objects are divided into three categories. The objects of the first group, presented in unshaded rectangles, control the AutoCAD interface and are common among all drawings. Objects in the second group, presented in lightly shaded rectangles, aid users in controlling the drawings, however do not have a visual realization. These objects have a common set of methods, properties, and one event, represented by a generic object called *AcadObject*. Objects in the third group, darkly shaded in Figure 5.2, are the graphical objects that are employed to form the visible part of the drawing. These objects inherit the properties, methods, and event of the generic *AcadObjects*. In addition, they share a common set of properties and methods represented by the *AcadEntity* object (Sutphin 1999). Sets of objects that can form a group are placed together into a *collection*. The object model in Figure 5.2 illustrates the relationship among collections and objects.

The *application* object is the ancestor of all AutoCAD objects and all other objects are, directly or indirectly, accessible only via application. *Document* object represents an AutoCAD drawing. The *activeDocument* property of the *application* object is used to

reference the current drawing. When programming from within AutoCAD, *thisDrawing* keyword can replace *application.activeDocument*. The document object is the parent object of *blocks*, *modelSpace*, *paperSpace*, *layers*, and *selectionSets* collections. The first three are used to access graphical objects, while the last two are used to control drawing style.

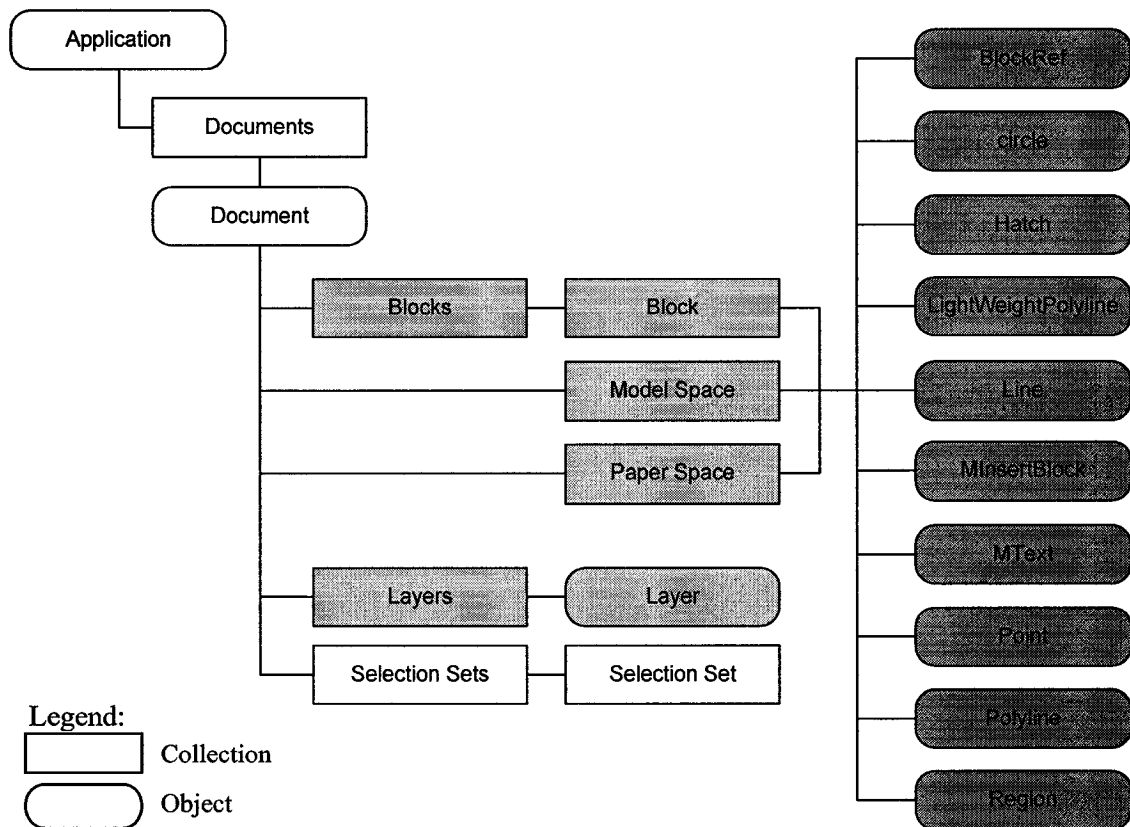


Figure 5.2. Partial object model of AutoCAD (Autodesk 2001)

Model space and paper space are two drawing spaces represented by their respective collections. Model space is usually where the user designs the drawing, and paper space is where the drawing layout is formatted for output. Graphical objects can as well be

added to *block* collection. A drawing may have several *block* collections, contained in the *blocks* collection. Block collections are used to collect AutoCAD entities into a single container. This container becomes a new drawing entity itself which can be inserted into the model space, paper space or other block definition. Manipulating the state of layers (e.g turn on/turn off) facilitates the management of the complex drawings. In CASL, layers are used to organize the graphical objects into logical groupings. *Layer* object, contained in the *layers* collection, represents these groupings of graphical objects.

5.2.2 Employed Graphical Objects

A list of AutoCAD graphical objects used in the implementation of CASL can be viewed in Figure 5.2. In this section three of these objects, namely *polyline*, *region*, and *blockRef* and *block* that are broadly used in the implementation of CASL are described. Having a view of the structure of these objects and their properties and methods is essential for understanding the implementation of the model.

To reference an object directly, it should be included in the calling hierarchy. For example *thisDrawing.ModelSpace.lineObject* gives access to a line object generated in the model space. To access the *application* object which is situated above all objects in the hierarchy, the *application* property of the *document* object is used. This property provides a link to the *application* object and its methods and properties. For example, *ThisDrawing.Application.Update* updates the application object. CASL uses the model space as the working area for generating objects and processing analysis. All graphical objects are generated in *modelSpece* collection via *thisDrawing* and using the generic

“addObject” method. For example the following statement adds a lightweight polyline to the model space.

ThisDrawing.ModelSpace.AddLightweightPolyline(VerticesList)

- **Polyline Object**

Table 5.1. Polyline object properties

Property	Description	Return Data Type
Area	Specifies the enclosed area of the polyline	Double
Closed	Specifies if the polyline is open or closed	Boolean
Color	Specifies the color of a polyline	integer
Coordinate	Specifies the coordinate of a single vertex in the polyline	Variant (array of doubles)
Coordinates	Specifies the coordinates for all vertices in the polyline	Variant (array of doubles)
Handle	Gets the handle of a polyline	String
Layer	Specifies the layer for a polyline	String
Linetype	Specifies the linetype of a polyline	String
LinetypeScale	Specifies the linetype scale of a polyline	Double
ObjectID	Gets the object ID of the polyline	Long
ObjectName	Gets the AutoCAD class name of the polyline	String
Thickness	Specifies the distance a polyline is extruded above or below its elevation	Double
Visible	Specifies the visibility of a polyline	Boolean

The AutoCAD entity that is used in CASL to generate the geometry of site and construction objects is *Polyline object*. Polyline is a 2D or 3D line composed of straight line and arc segments. The properties and methods of the *Polyline object* that were employed in the implementation of CASL are presented in Tables 5.1 and 5.2. A list of

complete properties and methods of the *Polyline object* can be found in appendix II. The *handle* property is of special importance. It indicates a unique identification for individual objects. It is a read-only property, selected randomly and assigned to an object upon its creation by AutoCAD. Since the handle of an object is a permanent characteristic for the lifetime of that object, it is the best way to reference an object.

Table 5.2. Polyline and region object methods

Method	Description	Return Data Type
Boolean*	Performs a Boolean operation between the object and another region object	
Copy	Duplicates the given object to the same location	Object
Delete	Deletes a specified object	
Explode	Explodes the compound object into sub-entities	Variant (array of objects)
GetBoundingBox	Gets two points of a box enclosing the specified object	Variant (three-element array of doubles)
GetXData	Gets the extended data (XData) associated with an object	
Highlight	Sets the highlight status for the given object, or for all objects in a given selection set	
IntersectWith	Gets the points where one object intersects another object in the drawing	Variant (array of doubles)
Move	Moves an object along a vector	
Offset**	Creates a new object at a specified offset distance from an existing object	Variant (array of objects)
Rotate	Rotates an object around a base point	
ScaleEntity	Scales an object equally in the X, Y, and Z directions	
SetXData	Sets the extended data (XData) associated with an object	
Update	Updates the object to the drawing screen	

* Applies only to region object

** Applies only to polyline object

- **Region Object**

Region object is a bounded planar face consisting of lines, polylines, circles, arcs, elliptical arcs, and curves. A set of latter objects, conforming a closed loop, should be created first and then converted to *region* object. The following statement is used in CASL to generate *region* objects:

ThisDrawing.ModelSpace.AddRegion(ObjectList)

ObjectList is the array of objects forming the closed coplanar face to be made into a *region*. Each object in the region retains its layer, linetype, and color. AutoCAD deletes the original objects after converting them to *regions*. *Region* objects are widely used in CASL, mainly due to the usability of the *Boolean* method. This method is specific to *region objects* and *3DSolids* and allows creating composite regions from the intersection, union, or subtraction of a region from two regions. This feature is specifically important in CASL, since many subtractions and intersections take place in the site analysis stage. This is discussed in more details under the developed functions in CASL. As such, in CASL all the geometries, including site and construction objects, site boundary, and site segments, are eventually converted to *region*. The signature for the Boolean method is as follow:

regionObject1.Boolean (Operation, regionObject2)

in which the first region object (*regionObject1*) is the one that *boolean* method applies to, and the second (*regionObject2*), is the object against which the operation is performed. The operation can be union, intersection, or subtraction. Other methods of the *region* object that were employed in CASL are presented in Table 5.2. Table 5.3 contains a list

of mostly used properties of *region* object. A complete list of region object methods and properties can be found in Appendix II.

Table 5.3 Region object properties

Property	Description	Return Data Type
Area	Specifies the enclosed area of the region	Double
Centroid	Gets the center of area or mass for a region or solid	Variant (array of doubles)
Color	Specifies the color of a region	Integer
Handle	Gets the handle of a region	String
Layer	Specifies the layer for a region	String
Linetype	Specifies the linetype of a region	String
LinetypeScale	Specifies the linetype scale of a region	Double
ObjectID	Gets the object ID of the region	Long
ObjectName	Gets the AutoCAD class name of the region	String
Perimeter	Gets the total length of the region loops	Double
Visible	Specifies the visibility of a region	Boolean

Block Object and BlockReference Object

Block object is a definition containing a name and a set of objects. Different types of block objects exist. The type of *block* object used in the implementation of CASL is simple block. A simple block is a collection of objects associated together to form a single object, known as block definition. It can be inserted, scaled, rotated, and exploded into its component objects. Simple blocks can be defined from geometry in the current drawing, and generate another AutoCAD drawing. This feature is used in CASL to store a copy of the *polyline* generated for a new site or construction object, in a separate AutoCAD drawing. Inserting an instance of a simple block into the current drawing creates a *blockReference* object. In CASL, *blockReference* objects are exploded upon

their insertion on the drawing into the original *polyline* object. CASL takes advantage of *sendCommand* method to write and insert blocks. This method sends a command, as a string, from VBA to be process in the current drawing. As such, the readily available “Write block” and “Insert” dialog boxes of AutoCAD were used for creating and inserting blocks, respectively. Thus, the properties and methods of *block* and *blockReference*, except for the *explode*, are not used in the code. However, a list of these properties and methods are brought in Appendix II. Alternatively, blocks can be written and inserted using the *Wblock* and *InsertBlock* methods as follow:

ThisDrawing.Wblock FileName, SelectionSet

ThisDrawing.InsertBlock(InsertionPoint, Name, Xscale, Yscale, ZScale, Rotation)

5.2.3 Object Implementation in CASL

Based on the design presented in chapter 3 for site, construction, and constraint objects, the three tiers of objects have been implemented as database entities, with their properties being the attributes of the tables. Figure 5.3 illustrates the implemented attributes and the relationships among entities. It should be noted that the conceptual design attributes presented in Tables 3.1 through 3.3 have been adjusted in the implementation phase.

In Figure 5.3, in each table, the *ID* attribute refers to a unique identification assigned to objects by the database upon the creation of the record. This attribute acts as the primary key, by which the relationships among tables are defined in a relational database. *Name* holds the user assigned name of the object. A descriptive name for an object facilitates

easy identification of that object. *X,Y,Z Coordinates* of the site objects are expressed relative to a site boundary, the coordinates of the bottom-left corner of the bounding box of which are (0,0,0). Construction objects do not have coordinates attributes since their location is not known until they are located. Once located, their location can be read from the drawing. In this implementation, the *height* attribute is put under the layout attributes. Alternatively the *thickness* property of *polylines* can be used for implementation. *Arrival* and *departure dates* are based on first day of project (00/00/00). These attributes are presently used for generating the queue of construction objects, however, they are as well predicted for the expansion of model to incorporate the time factor. When the duration of an object on site is finished, the space it was occupying will be free to accommodate the succeeding objects. *Mobility* specifies if the object is mobile, or stationary. *Movability* indicates if the object's location can be changed after it has been located, and *cost of relocation* indicates the desirability of moving the object on an increasing scale of 1 to 5. *Orientation* attribute specifies the rotation angle at the time the object is inserted from the drawing file into the layout. *Flexibility* attribute indicates if the shape of the object can be changed and takes one of the three values: flexible, sizable, or rigid. For sizable objects, *smallest unit* requires a value. *Containability* attribute of an object indicates if other objects can be located inside it. *File path* holds the address of the AutoCAD file containing the geometry of the object. Depending on the *relation* selected for the constraint object, the *distance* attribute serves a different role. For linear relationships (polar and linear closeness) this attribute holds the offset distance and for distance relations (within x distance, not within x distance) it holds the length of x (see Table 3.4).

Site and *construction handles* record the handle of the geometry of objects as they are generated in AutoCAD.

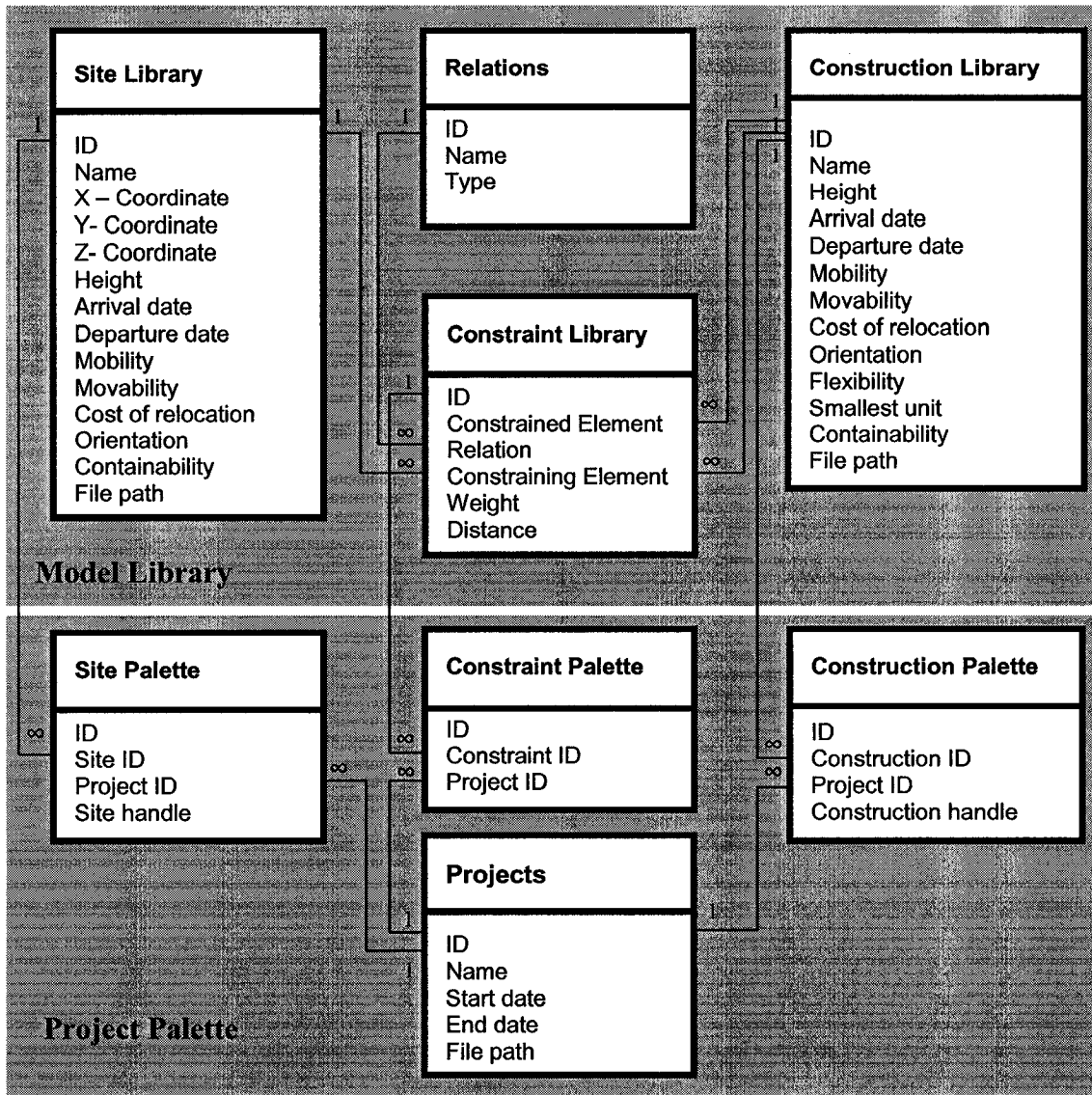


Figure 5.3. Relationship diagram of the database

5.2.4 Developed Functions in CASL

Different functions were implemented to perform the functionalities designed for CASL. Three of these functions, namely *rings*, *interSub*, and *discretization*, which play a fundamental role in the implementation of the model, are discussed in this section.

- **Rings Function**

This function is used to generate graphical representation for relations used in defining the constraint objects. Of the six types of relations described in chapter 3, one case, regarding the polar closeness relations, is described here. This section of function generates concentric rings round the constraining element (Figure 5.4). The function starts by drawing a circle circumscribing the constraining element using *addCircle* method. Using the *offset* method, the circle is offset several times, until the whole site area is covered (Figure 5.4a). An extra copy of every offset circle is made using the *copy* method, to be used later in the subtraction procedure (see below). The offset and copied circles are sent to an array to be converted to *region objects*. Once converted to *region objects*, *Boolean* methods can be performed on them. Consecutive rings are subtracted from each other to create rings (Figure 5.4b). Finally, the rings are trimmed from the edges of site boundary using subtraction operation of *Boolean* method (Figure 5.4c). The following pseudocode summarizes the described section of *rings* function:

OffsetNumber: Max distance from center of constraining element to corners of site/offset distance

For n = 1 to offsetNumber

Offset the circle by offset distance * n [circle.offset (offset distance * n)]

Copy the new circle [circle.copy]

Send the offset and copied circles to an array of objects

Convert the array of circle objects into array of region objects [*regionArray*]

Subtract consecutive region objects to create rings

[For n = 1 to (offsetNumber* 2) - 1 Step 2

regionArray(n).Boolean acSubtraction, regionArray(n + 1)]

Trim the rings from the site boundary [*siteObj*]

For n = 1 To Upper Bound of (regionArray) - 1 Step 2

Check if rings intersect with site boundary [*regionArray*(n).IntersectWith (*siteObj*)]

Make a copy of site boundary [*siteCopy* = *siteObject*.Copy]

Subtract the site boundary from the ring

[*regionobject*(n).Boolean acIntersection, *siteCopy*]

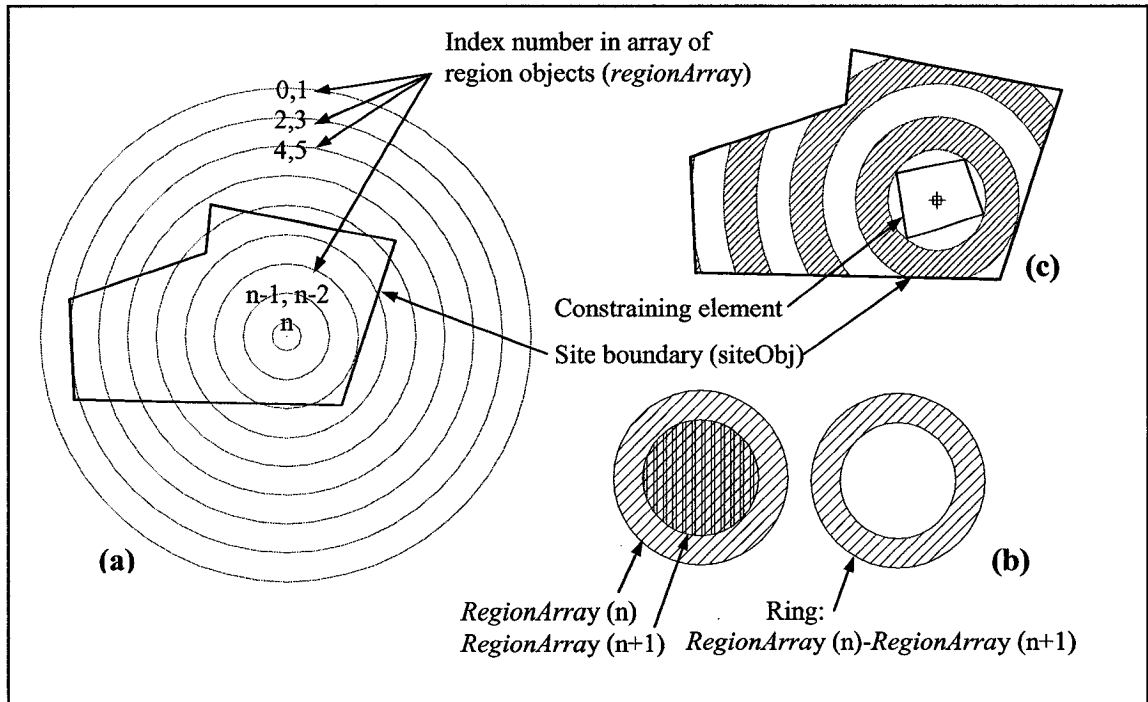


Figure 5.4. *Rings* function (a) Offsetting central circle, copy offset circles, send them to array of circles, convert to array of region objects (b) Subtracting consecutive circles (c) Subtracting the rings from site boundary

- **InterSub Function**

This function takes two overlapping *region* objects as arguments, cuts them through the overlapping edges, and returns the resulted *region* objects, as shown in the input and output of Figure 5.5. To implement the *interSub* function Boolean method is used to alter *region* objects utilizing subtraction, and intersection operations. The Boolean operations take place between two objects at a time; the first object, “calling object”, calls the method and the second, “passed object”, is passed as argument. When performing

Boolean operations, the calling object is altered according to the operation used (i.e. subtraction, intersection, or union), and the passed object, along with its handle, is destroyed. In other words the outcome of the Boolean operation inherits the *handle* property of the calling object. As such, the role objects perform in the operation, i.e. whether they are calling objects or passed objects, affects the outcome geometry of the *Boolean* method as shown in Figure 5.5. For example, the following statement:

ObjectA.Boolean (acIntersection, ObjectB)

denotes an intersection operation called from object A, passing object B as argument.

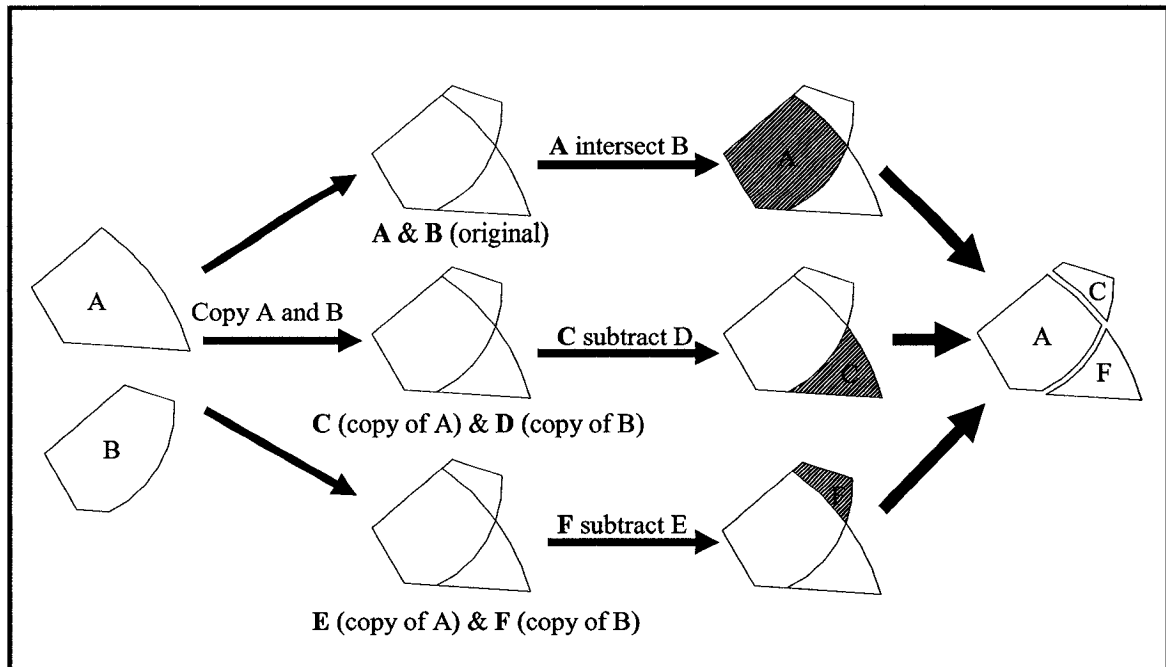


Figure 5.5. *InterSub* function

The intersection area, highlighted in the first line of Figure 5.5, inherits the handle of the calling object (i.e. object A), and the passed object (i.e. object B) is eliminated. However, in order to be able to perform the geometric reasoning in CASL, the remaining of the

original objects has to be captured, as shown in the final result depicted in Figure 5.5. To obtain this outcome, two copies of the original objects are generated as objects C-D and E-F, and the subtraction operation is performed for each of the two sets of copies. This operation is once called from the copy of object A (i.e. object C), and then called from the copy of object B (i.e. object F). Consequently, the three generated objects inherit the handles of objects A, C, and F, and when put together, yield the *region* objects shown at the right hand side of Figure 5.5.

- **Discretization Function**

When analyzing site space for locating a construction object, *region* objects representing different constraints assigned to the object in hand are generated using the *rings* function. The *discretization* function takes all of these *region* objects and performs *interSub* function among all pairs, until no two overlapping *regions* are remained. As a result the site is divided into smaller segments, represented by new *region* objects.

All the existing *region* objects are selected and stored in a *selectionSet* object. The number of objects in a selection set can be counted via its *count* property. Two nested *for* loops are created to perform the *InterSub* function among all pairs of objects. Every time the *InterSub* function is performed, two new objects are created that are not part of the original objects (e.g. object C and F in Figure 5.5), and one of the original objects is eliminated (e.g. object B in Figure 5.5). At the end of the *for* loops, upon application of *InterSub* function to all pairs of objects in the *selectionSet*, a number of new objects are created and a number of objects from the *selectionSet* are destroyed. At this point, the

model re-selects the current set of *region* objects and places them, in the *selectionSet*, and in a similar manner, applies the *InterSub* function. This function takes place several times through a *Do-While* loop until there are no two intersecting *region* objects left. A *flag* indicates if no more intersection can take place between any pair of objects. The following pseudocode summarizes the *discretization* function.

```

Do
  Select all current objects into a selectionSet
  Flag = False
  For n = 0 to selectionSet.Count
    If object n of selectionSet still exists
      For m = 0 to n-1
        If object m of selectionSet still exists
          If n-m pair is not in nonOverlap
            If object n intersectWith object m
              Perform InterSub between object n and object m
              Add pairs to nonOverlap
              Flag = True
            Else add n-m pair to nonOverlap
          Loop While Flag = True

```

Before performing the *interSub* function between a pair of region objects, first it is examined if they still exist, i.e. have not been destroyed through Boolean method in the previous executions of *interSub*. Also, it is examined whether the two selected regions are overlapping, through *intersectWith* method (see Table 5.2). To optimize the speed of performance, a collection named *nonOverlap* is created to register the pairs that do not overlap. Any pair that has already been examined and did not have an intersecting area, as well as pairs that have been resulted from *interSub* function, are added to this collection and not examined again. For example in Figure 5.5, after the *interSub* function

has taken place, A-C, A-F, and C-F pairs are added to *nonOverlap*. Furthermore, any descendants of a *nonOverlap* pair will not be examined for the intersection either. For example assume region G overlapping with region A of the previous example as shown in Figure 5.6. The *interSub* function is performed between G and A regions, and as a result region A is destroyed and regions H and I created. The new *nonOverlap* pairs (i.e. G-H, G-I, and H-I) are added to the collection. As well, since regions H and I are descendants of region A, they replace A in the previous set of pairs (i.e. A-C, A-F, and C-F). The updated *nonOverlap* collection is shown in Figure 5.6.

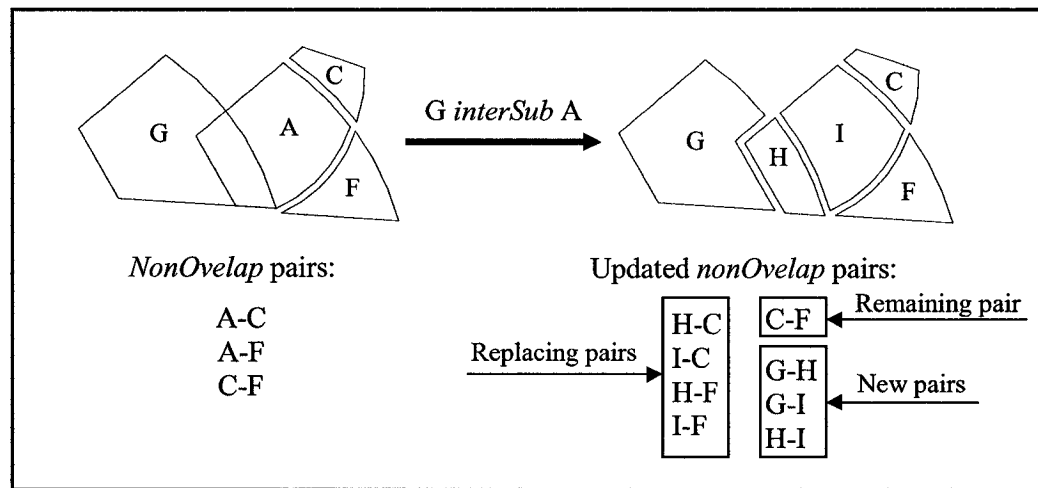


Figure 5.6. Updating *nonOverlap* pairs

5.3 IMPLEMENTED MENUS

This section describes CASL implementation, along with its forms and their functionality. CASL includes two main menus: project setup and layout menus. These menus represent the implementation of the two main modules in the proposed model; i.e. Project and Layout Control Modules, respectively. Special attention has been given to the

development of the user interface to yield a user-friendly and “usable” tool. Usability of a system is defined as its accessibility through interaction with interface, features, and structure. Such system supports user’s activities and their flow, while adding new concepts and automating steps in the process. As such, a usable system supports the way practitioners’ reason and process their work (Holtzblatt and Jones 1993).

Typographical conventions

In this chapter, specific terms are set in typefaces to enhance readability of the text. Throughout the remaining of this chapter the typographical conventions listed in Table 5.4 are used for the description of the prototype.

Table 5.4. Typographical conventions

Text element	Typeface	Example
Control object on the forms (e.g. command button, option button, check box, frame, tab strip, and grid)	<ul style="list-style-type: none"> ▪ Sentence case ▪ Inside quotation marks 	“Define construction objects” “Explode”
Form name	<ul style="list-style-type: none"> ▪ Title Case ▪ Inside quotation marks 	“Define Construction Objects” “Object Properties”
Method	<ul style="list-style-type: none"> ▪ Upper and lower case 	<i>insertBlock</i>
Function	<ul style="list-style-type: none"> ▪ Concatenated ▪ Italic face 	<i>interSub, discretization</i>
Property	<ul style="list-style-type: none"> ▪ Lower case 	<i>closed</i>
Attribute	<ul style="list-style-type: none"> ▪ Italic face 	<i>weight, distance</i>
AutoCAD command	<ul style="list-style-type: none"> ▪ Lower case 	<i>wblock</i>
AutoCAD object	<ul style="list-style-type: none"> ▪ Concatenated ▪ Italic face 	<i>block, region, polyline</i>

5.3.1 Project Setup Menu

Figure 5.7 shows the “Project Setup Menu” used in setting up of a new project in CASL. The five command buttons, implement the five steps of project setup phase presented in Figure 4.6; i.e. creating site boundary, defining site objects, defining construction objects, defining constraint objects, and queuing construction objects. At the beginning of the project setup, only “Create site” command button is enabled, to ensure creation of site boundary prior to other objects. Creating a site boundary enables other buttons in “Project Setup Menu” form.

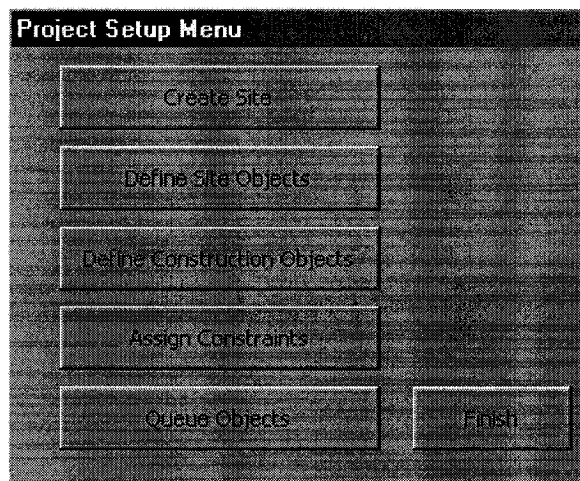


Figure 5.7. “Project Setup Menu” form

To generate the geometry for site boundary, site objects, and construction objects, the user can choose one of the options provided in “Select Drawing Style” form (Figure 5.2):

1) “Insert from file” is used when the user has the geometry of the object in an AutoCAD drawing file. This option inserts the drawing file as a *block object* using the *InsertBlock* method. Selecting this option activates the “Insert” dialog box of AutoCAD shown in Figure 5.9. This form allows the user to browse for the file, indicate the insertion point,

scale, and rotation angle of the object on screen. The “Explode” check box should be selected to convert the inserted *block object* into *polyline*.

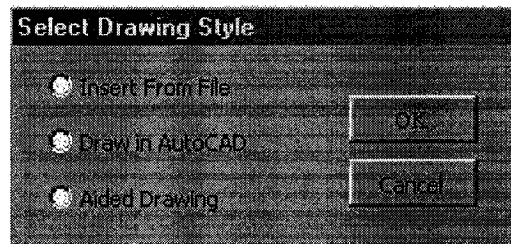


Figure 5.8. “Select Drawing Style” form

2) “Draw in AutoCAD” option is used when the object is not readily available in a file, but the user knows how to operate in AutoCAD environment. Upon the selection of this option, the model passes the control to AutoCAD, prompting the user to draw the graphical object as a *polyline object*. When the drawing is finished the model resumes control by capturing the *polyline object* and ensuring that it is bounded through its *closed* property.

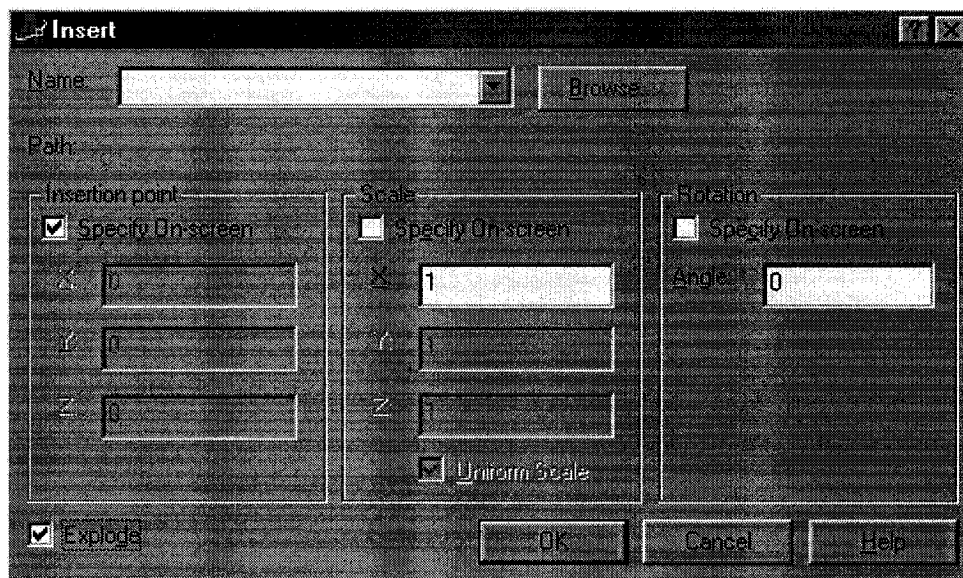


Figure 5.9. “Insert” form from AutoCAD

3) “Aided drawing” is used when the object is not readily available in a previously drawn file, and the user is not familiar with working in AutoCAD. Selecting this option activates the “Aided Drawing” form (Figure 5.10). The user is required to input the geometric data in text format. This information is either in the form of the coordinates of the corners of the object (vertices), or the length of each segment and the angle it makes with the next segment. Each segment can be defined to be a line or an arc. The selection can be made via the two tab strips on this form. To input a new vertex /segment the “New” command button is selected to provide new row in the table. Any vertice/segment can be removed using the “Delete” command button. Once all the vertices/segments are entered into the table, the “Draw polyline” command button is used to create the geometry of the object by *AddPolyline* method.

Point #	X	Y	Line/Arc
Vertex 1	129	160.5	line
Vertex 2	141	205	line
Vertex 3	241	233.7	line
Vertex 4	343	156.5	line
Vertex 5	270	67.8	line
Vertex 6	165	72	Arc
Vertex 7	161.8	155	Arc
Vertex 8	129	160.5	line

Figure 5.10. “Aided Drawing” form

Once the polyline representing the site boundary is generated, the coordinates of its corners are retrieved using *coordinates* property, and stored in a variable (*coord*) that is accessible for other subroutines. The site is then converted into a *region object*. At this point other polylines, such as islands or extensions, can be deducted from or added to the site region to shape the final site area.

Once the site boundary is defined, the model returns to “Project Setup Menu” (Figure 5.7). At this point “Define site objects” and “Define construction objects” command buttons are enabled. Selecting them activates “Define Site Objects” and “Define Construction Objects” forms respectively, which are similar in form and function. Figure 5.11 demonstrates the latter form. The list on the left provides a gallery of construction objects that exist in the construction library. The user can select objects from the library and add them to the list on the right, representing the construction palette. When an object is added to the project palette, an instance of its geometry is generated on site using the *Insert* method and *file path* property. The generated object is a *block object*, and its *handle* property is added to PP for future access. For site objects, the insertion point is read from database and the object is inserted in its position on layout. Since the location of construction objects is yet to be determined, they are inserted on the side of the layout. “Remove from palette” command button removes selected objects from the construction palette list, and from the drawing, using the *delete* method. The layout and CAD properties of objects can be viewed using the “View properties” command button (Figure 5.12). The “Object Properties” form is similar for the site and construction objects. However, the *flexibility* and *smallest unit* properties are not applicable for site objects,

and so is the *location* property for construction objects. Note that the thumbnail preview of the CAD drawing in Figure 5.12 is not implemented in the current code, but can be added using the pre-defined *thumbnail object*.

Define Construction Objects

Construction Library	Construction Palette
Factory	Lorry park
Factory parking	Office
Gatehouse	Parking
Lorry park	
Office	
Parking	
Garage	
Storage	
Welding shop	
Plumbing shop	
Electric shop	
Formwork storage area	
Steel storage area	
Electrical storage area	
Equipment parking	
Lab parking	

Buttons: Add to Palette >>, Remove from palette, View properties, Modify object, Create new object, Finish

Figure 5.11. “Define Construction Objects” form

The “Modify object” command button allows user to modify the layout properties of site and/or construction objects after they have been added to site palette and/or project palette, respectively, through “Modify Site Object” and “Modify Construction Object” (shown in Figure 5.13) forms. The “Location” text box is not enabled for construction objects. As well, the “Smallest unit” text box is enabled only when the “Flexibility” is set

to “Sizable” option. For site objects the options regarding both “Flexibility” and “Smallest unit” are disabled. Modifications can be applied to the same object and change the layout properties of the object permanently, or they can be saved as a new object in the respective library.

Figure 5.12. “Object Properties” form

The “Create new object” command button on “Define Construction Object” form (Figure 5.11) allows user to create objects that are not readily available from the construction library. When creating a new site/construction object, the user is first prompted to generate the geometry of that object, using the forms shown in Figures 5.8 through 5.10. If the geometry is drawn in AutoCAD, or using the aided drawing option, the *wblock*

command of AutoCAD is used to convert the generated *polyline* to a *block object*, and at the same time write the block into a new drawing file. Figure 5.14 shows the “Write

Modify Construction Object

Object Name: Storage

Location:

Height: 5 m

Orientation: 0

Arrival date on site: 00/02/00

Departure date from site: 00/08/02

Movable: ☒ Yes ☐ No

Mobility: ☒ Mobile ☐ Stationary

Cost of relocation (1-5): 3

Containable: ☒ Yes ☐ No

Flexibility: ☒ Rigid ☐ Sizable ☐ Flexible

Smallest unit: ☐ X ☐ Y ☐ Z

Cancel Save as new object Modify object

Figure 5.13. “Modify Construction Object” form

Block” form. On the “Base point” frame, the “Pick point” is set to the centroid of the *polyline*. When the block is inserted into a drawing, this point is used as the base point of insertion. In the “Objects” frame, the “Delete from drawing” option is selected. This

option converts the selected object into a block and saves it as a drawing file. The location of this file is indicated in the “Location” text box of the “Destination” frame. This string is stored in the *file path* attribute in model’s database for the newly created object. Similarly, if the geometry of the object is inserted from a file (see Figure 5.9), the value of the “Name” text box is used for the object’s *file path* property. The *handle* of the newly created *block object* is recorded in the respective table of the Project Palette.

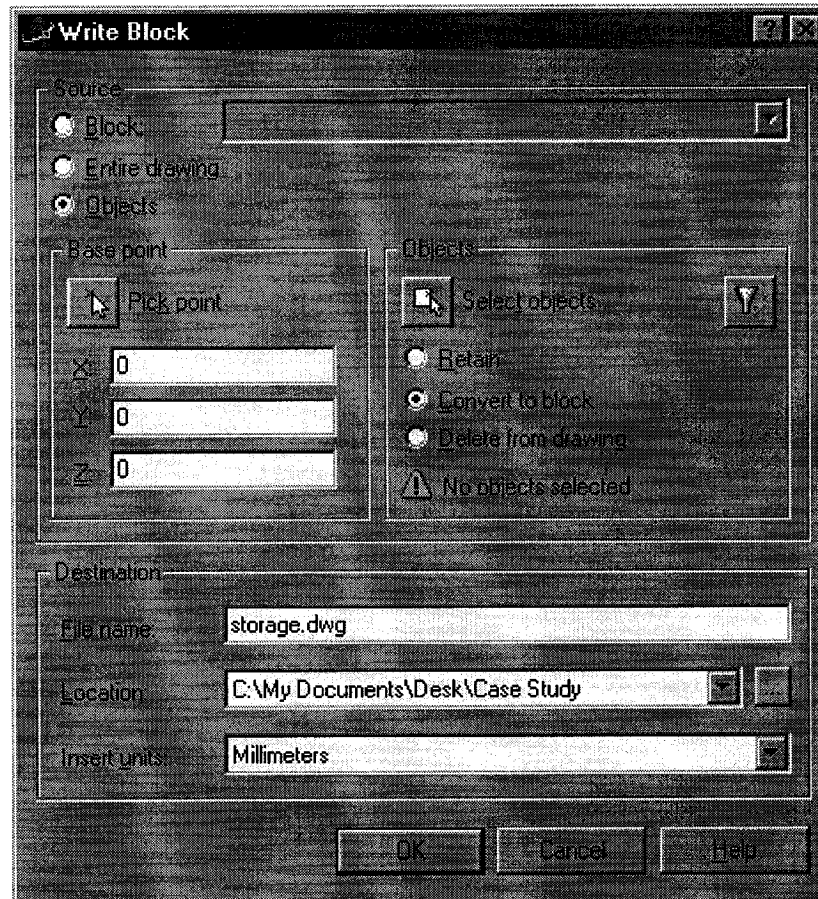


Figure 5.14. “Write Block” form from AutoCAD

Once the geometry of the object is drawn, the user has to input the information in the “Layout Properties” form, which is identical to the forms used for modification of layout

library of the database, the model performs a query for the constraint objects whose constrained element is the selected construction object. The user can select constraints from this list and add them to the “Constraint palette” grid, using the “Add to palette” command button. The model verifies if its constraining element is previously selected to the site palette or construction palette. If it is not, the user can choose to add the constraining element to PP, or to discard the constraint object. “Remove from palette” command button allows deleting records in the “Constraint palette” grid. The *weight* and *distance* attributes can be modified using the “Modify object” command button. This modification is permanently registered in the constraint library of the database. “New constraints” command button activates the “Assign Constraints” form, where user can create new constraints for the selected construction object (Figure 5.16).

Relation	Constraining Element	Weight
close to	Temp Office	90
west of	Reinforcement St	85
far from	General Store	20
	Concrete Batch Pl	
	Factory	
	Factory Parking	
	Gatehouse	
	Lorry Park	

Figure 5.16. “Assign Constraints” form

The selected construction object from the previous form, representing the constrained element, is shown on top left corner of the “Assign Constraint” form. The “Relation”

combo box retrieves available relationships from the relations table in Model Library. The “Constraining element” combo box is populated with objects from construction and site palettes. The user can create a new constraint object by selecting the suitable relation and constraining elements, and assigning a weight (between 1 to 100) to it in the “Weight” text box. Pressing “Add to constraint” command button adds the created constraint to the “Constraint palette” grid on this form. Pressing “OK” reactivates the “Select Constraint Objects” form and transfers the defined constraints for the selected construction object to the “Constraint Palette” grid of this form. Here, constraint objects for other construction objects in the list can be defined, in the same manner. Once all the required constraints for all the construction objects are defined, “Finish” command button reactivates the “Project Setup Menu” form (Figure 5.7). At this point the “Queue objects” command button is enabled and selecting it activates “Queuing Construction Objects” form (Figure 5.17)

Figure 5.17. “Queuing Construction Objects” form

This form offers a tool to set multi-attributed criteria for queuing objects, enabled by selecting the first option “Queue construction objects by:”. Selecting each of the attributes enables a text box next to it where the user can associate a weight to that attribute to indicate its importance in relations to others. “Construct queue” calculates the queuing score based on the selection and sorts the construction objects in the palette in descending order of their queuing score in the “Queue of construction objects” list. The user can change the location of the objects in the queue by selecting one and using “Move up” and “Move down” command buttons. The queue of objects can be reconstructed several times by altering the selected attributes and assigning different weights to them until it is deemed satisfactory. The “User-defined queue” option button disables the attributes and retrieves the list of construction objects according to the order they have been added to the construction palette. The user can arrange the order of objects in the queue, as well, using the “Move up” and “Move down” command buttons. Pressing “OK” reactivates the “Project Setup Menu” form (Figure 5.7). At this point all the command buttons are enabled and the user is allowed to re-select any, and perform modifications on the Project Palette. The “Finish” command button on this form indicates the end of project setup phase and activates the layout menus.

5.3.2 Layout Menu

Once the queue of construction objects is formed, the model starts to locate construction objects on site. The “Analyze site” command button analyzes site space for construction objects in the order they appear in the queue. The name of the construction object at hand, and its position in the queue is stated on top of the form. The constraints assigned

To aid the user in visually identify their ranks, each segment is assigned a color. AutoCAD contains 255 colors, identified by numbers from 1 to 255. Site segments ranked 1 to 255 are assigned the color relative to their rank. This provides a general view for the neighborhood of near-optimum locations. For more precise identification, regions can be identified based on their rank. An invisible list box, linked and parallel to “Site regions” grid, registers the *handle* property of site segments. This aids in tracing and addressing the site segments based on their rank and utility score. “Identify region” command button prompts the user to indicate a segment by its rank and highlights that segment using *AddHatch* method. As an alternative, the utility score of the site segments are registered as extended data (Xdata) in the *region object* using the *SetXdata* method. Double-clicking a site segment retrieves the utility score from the Xdata by *GetXdata* method and displays it in a message box.

If higher precision is required, “Refine search” command button allows to select an area on site and rerun the analysis in that area with smaller offset distance for the linear relationships. The implementation of this part is similar to that of initial creation of site boundary. The model prompts the user to draw a *polyline*, converts it to a *region object*, and performs the analysis in this area using *rings*, *interSub*, and *discretization* functions. Every time the *discretization* function is called, a new *layer object* is created to store the site segments resulting from the analysis. The name of the new layer consists of the name of the construction object for which the analysis is performed, concatenated with an index number. In refinement iterations of site analysis for the same construction object, the index number increases by unity. A new layer is created for the refined search and is

made the *current* layer. The previous layer is turned off to avoid visual confusion. Thus, the information regarding every search iteration is stored separately and can be later viewed aid user in deciding the final location of the object. “View layers” provides access to a list of created layers, each of which can be turned on and off as needed.

“Get score” command button allows user to study the utility score for a location other than those offered by the analysis; i.e. the centroid of site segments. Selecting this button activates the “Calculate Utility Score” form (Figure 5.19). “Select point” option calculates the total utility score for a point on site indicated by the user. This point can either be selected on screen, or indicated by its coordinates. “Select region” option retrieves this score for one of the existing regions, as an alternative to double-clicking the region mentioned earlier. Selecting “Draw region” option prompts the user to draw a free-shaped *polyline*, converts it to a *region object*, and calculates the score at its centroid.

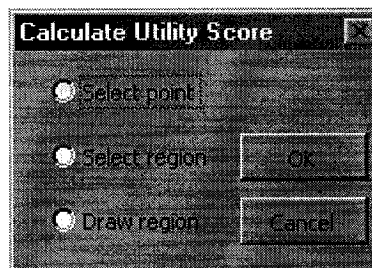


Figure 5.19. “Calculate Utility Score” form

Once the final location of the object is decided, “Locate object” command button is selected to place it on the layout. To allow user to indicate the selected location,

“Locating Construction Object” form is activated (Figure 5.20). The first option, allows the user to select one of the ranked site segments listed in the “Site regions” grid of the “Layout Menu” form (Figure 5.18). The default is to locate the object in the segment ranked first. However, the user can indicate other segments by their ranks in the provided text box. The second option allows the user to select a free point on the site, based on the analysis provided and the studies made on other locations on site.

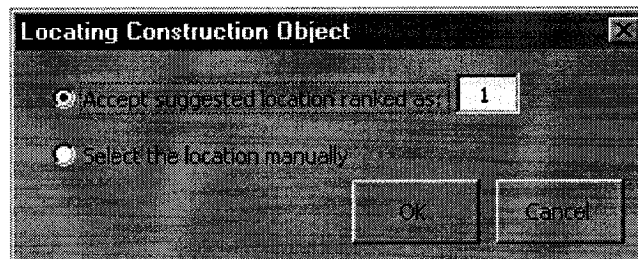


Figure 5.20. “Locating Construction Object” form

By indicating the location, the model places the *region object* representing the construction object at hand, matching its centroid with the selected location. At this point the “Refinement” form is activated (Figure 5.21). This form allows fine-tuning the final location of object on the layout and modifying its shape, if necessary. The “Adjacency” and “Alignment” command buttons activate menus similar to those shown in Figure 4.11. The user is prompted to select a base object from the layout, and then to select one of the adjacency or alignment options from the menu. These options indicate the desired position of bounding box of the object at hand, with regards to the bounding box of the base object. Accordingly, the model changes the location of the object using the *move* and *getBoundingBox* methods.

The “Rotation” command button prompts the user for a rotation angle. The model then revolves the object by the indicated angle round its centroid using the *rotate* method. “Free move” selects the located object on screen and passes the control to AutoCAD, allowing the user to use *move* command to finalize its location. By selecting “Change shape” command button, the model first verifies the *flexibility* property of the construction object. If it is set to *flexible* or *sizable*, the model allows for shape modification. The shape of *rigid* objects cannot be modified. The user is prompted to generate the new *polyline* using the *addPline* method, which is then converted to *region object*. Once the location and shape of the construction object at hand is finalized, the model updates the available site area by subtracting the *region object* of the construction object from that of site boundary, using the *Boolean* method.

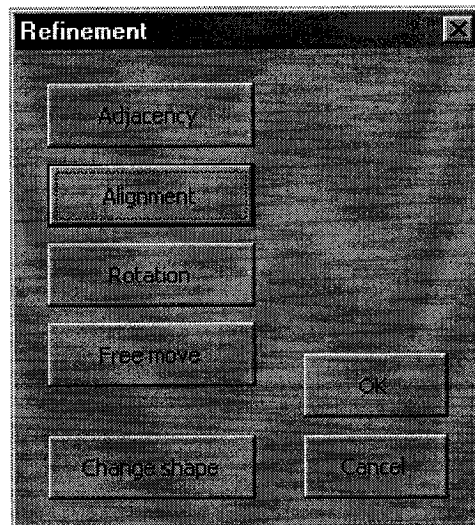


Figure 5.21. “Refinement” form

Selecting “OK” or “Cancel” ends the locating process for the object at hand and reactivates the “Layout Menu” form (Figure 5.18). The model verifies if there are more

objects in the queue list and repeats the locating process for each. Once all the construction objects are located, the “Evaluate layout” command button from the “Layout Menu” is enabled and the user can get the total utility score (U_t) for the layout. The value of this score is calculated as described in section 4.5.3. If the result is not satisfactory, the user can select “Project setup menu”. This command button reactivates the “Project Setup Menu” form where the settings of the project such as weights and order of objects in queue can be modified to re-perform the layout process. The layout process can be repeated as required until the layout is deemed satisfactory by the user. The “Exit” command button terminates the program.

5.4 SUMMARY

This chapter presented CASL, an implementation of the developed site layout model. CASL is implemented using VBA environment of AutoCAD and employing Access as model’s database. The main advantage of VBA over other programming interfaces of AutoCAD is that it runs in-process with AutoCAD, which results in fast execution of the program. As well, dialog construction is quick in VBA, which aids in fast prototype development. VBA also provides flexibility in the development of application by allowing them to be standalone or imbedded in drawings (Autodesk 2001). As such, on one hand, through ActiveX technology, VBA gets access to programmable objects of AutoCAD, and on the other hand, through SQL commands, can perform data manipulation in the Access database.

AutoCAD object model and the hierarchy among its objects were briefly presented. Three AutoCAD objects that were essential in the implementation of CASL, along with their methods and properties were described. The implementation of CASL objects as entities in Access environment was explained and their implemented attributes was enumerated. Through *handle* property, a dynamic link from each graphical object to the Project Palette is established so that the layout attributes stored in the Access database can be accessed and modified from within AutoCAD. In the end the functionality of CASL was described through its user interface and developed forms. Special attention was given to the development of user interface, to yield a user-friendly, and hence, a usable tool.

CHAPTER 6

CASE STUDIES AND PERFORMANCE EVALUATION

6.1 INTRODUCTION

In this chapter the performance evaluation of the developed model is presented. Three case examples are analyzed using CASL, to illustrate its functionality and usability. The examples start with a simple case and increase in complexity. The first example is a case of *location allocation* for single dimensionless object, in which only closeness constraints are considered. The second example contains multiple facilities, yet, only considers closeness constraint. These two examples were drawn from literature to enable a comparison with models developed by others. The third example is an actual project, developed and analyzed with the aid of the project's site engineer. This problem consists of multiple facilities with irregular shapes and considers a variety of locating constraints. The generated layout is then presented to the site engineer and his judgment and remarks are reported.

6.2 CASE EXAMPLE 1: SINGLE FACILITY, SINGLE CONSTRAINT

This case example essentially seeks the best location for a water fountain on the corridors of a manufacturing facility, so as to minimize the overall travel distance of employees between their offices and the water fountain (Zouein 1996). Figure 6.1 shows the layout of the facility with 20 departments, housing a range of employees. The numbers inside the parentheses indicate the number of employees in each department.

Mapping the problem into CASL, the departments play the role of *site objects*, while the water fountain is the single *construction object* to be located. The *constraint objects* for this problem include closeness of the water fountain to each department, defining the objective as the minimum total weighted distance. The weight of each closeness constraints is determined by the number of employees in the associated department. Out of the twenty departments, only seven departments, highlighted in Figure 6.1, accommodate employees and are considered in the location process. The rest of the departments, such as shipping or storage departments, do not have resident employees, and hence the closeness weight assigned to them is null. The shaded area in Figure 6.1 represents the corridor area, which defines the solution space (i.e. available site area). Data pertaining to the involved departments is summarized in Table 6.1.

Table 6.1. Involved departments and related data (Zouein 1996)

Number	Name of department	Number of employees	Coordinates	
			X	Y
1	Offices	90	49.54	43.68
2	Small part storage	12	160.50	62.00
3	PCB Manufacturing	34	34.96	138.98
4	Dept. 1-386	44	212.50	205.50
5	Dept. 2-386	60	239.40	155.73
6	Dept. 3-486	25	62.00	179.00
7	Dept. 4-486	30	114.50	205.50

It is assumed that all the employees are uniformly distributed in each department. Hence, distances are measured from the geometric centroid of each department. Also, since the problem involves locating a single facility (i.e. location problem), the dimensions of the fountain are ignored. These assumptions are identical to those introduced by Zouein (1996), so as to enable a comparison.

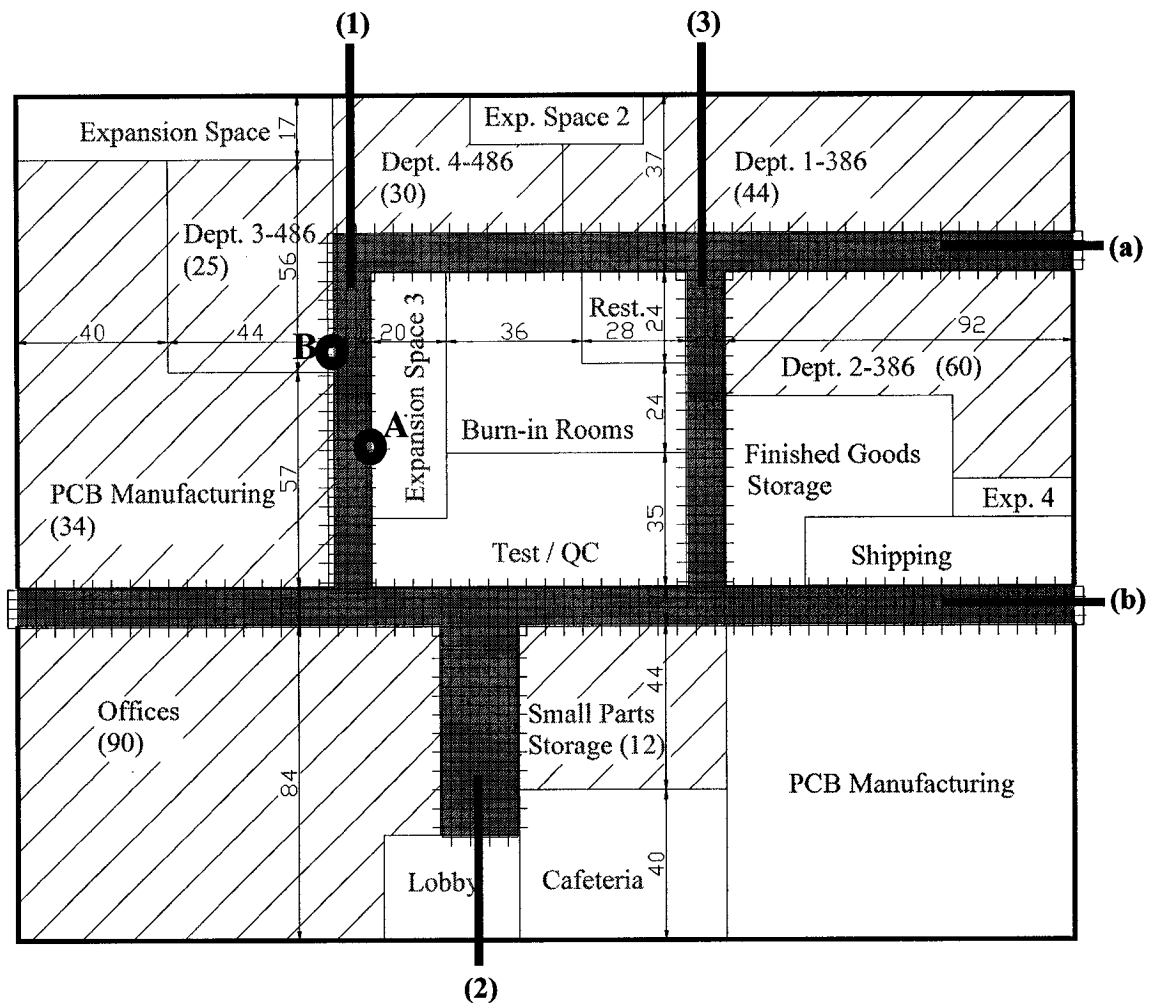


Figure 6.1. Layout of the manufacturing office and analyzing grid

The first step is to set the project palette. The site area being analyzed for this problem consists of the corridor. The departments are previously generated and stored in the database. In the Project Module, the involved departments are selected and added to the site palette. Water fountain is the only object selected into the construction palette. Seven constraint objects are assigned to this object with closeness being the relation and each involved department from Table 6.1 being the constraining element. Once the problem is defined, the project palette is sent to the layout control module to find the optimum location. The site analysis for this problem is performed twice; once using a standard rectilinear grid, and once using the site discretization method proposed in chapter 3. Both results are compared with the one reported by Zouein.

In the first analysis, the grid system is generated for the site boundary (i.e. corridor), as shown in Figure 6.1. The utility score (U) is measured for each grid point using Equation 3.15. Since the objective for this problem is to minimize the total weighted distance, this equation can be rewritten as:

$$U = \sum_1^n (d - d_i) \cdot W_i \quad 6.1$$

in which d is the longest dimension of the site boundary; d_i is the distance between the centroid of the i th department and the grid point being considered; and W_i is the weight of the closeness relationship between the fountain and the i th department. However, to enable a comparison with the results of Zouein (1996) and to apply similar measuring system, for this example the utility score is measured in reverse (e.g. ascending) order. As such, Equation 6.1 is replaced by:

$$U = \sum_1^n d_i \cdot W_i \quad 6.2$$

Once the utility function for all grid points are calculated, they are ranked in ascending order and locations with the minimum total travel distance are identified. Both rectilinear and Euclidean distance measurements were used to analyze this example. The points identified as A and B in Figure 6.1, depict the best location when using Euclidean and rectilinear measurements, respectively. Table 6.2 includes the coordinates and the u values of points A and B. Finer grids around points A or B can be generated until the desired precision is reached. Location B is the same as that found in Zouein (1996), where the rectilinear measurements were used as well ($X= 84.05$, $Y= 155.73$, $U= 3145.28$). The slight difference in the location of point A can be associated with employing the Euclidean measurements. While for this specific problem, in which the movements are performed in corridors, it is more appropriate to use rectilinear distances, the author believes that the path taken in open spaces such as construction sites is closer to Euclidean distances.

Table 6.2. Comparison of results using Euclidean and rectilinear measurements

Point	X coordinate	Y coordinate	u value	
			Euclidean	Rectilinear
A	94	130	30802.34	39385.94
B	84	155	31745.82	38153.51

Figure 6.2 shows a 3D graph of the corridor area when analyzed using Euclidean measurement. The shape of the corridor is rendered in X and Y axis, while Z axis

represents the utility score for each point. To enhance readability, corridors identified as 1, 2, 3 and a, b in Figure 6.1 are marked in Figure 6.2. The arrow in this figure points to the area with the minimum total weighted distance (point A in Figure 6.1).

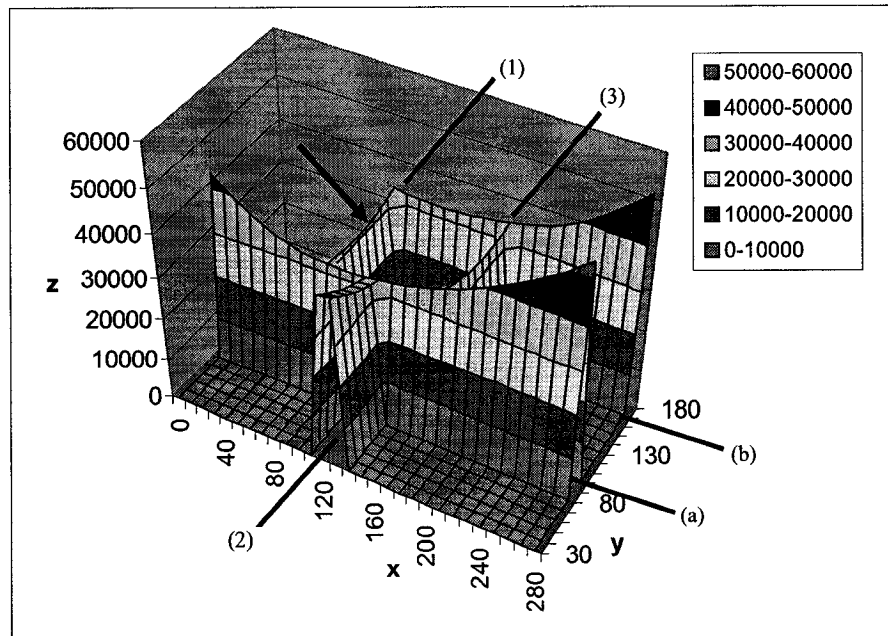


Figure 6.2. Total weighted distance graph for the corridor area

Figure 6.3 shows the space analysis for the corridor area using the proposed graphical discretization method. The closeness rings are generated from the centroid of each department. The corridor is divided into segments and each is assigned a utility scores using Equation (6.1) measured in Euclidean distance. It took 30 seconds for the model, run on a Pentium IV processor, to discretize site into 74 segments, measure their utility function, and rank them. As such, the area marked C gets the highest utility score. Points A and B from the grid analysis are also shown in Figure 6.3. As it can be inferred from the figure, area C encompasses the point A and is very close to point B. The offset

distance for the closeness rings are chosen 50 ft at the first iteration. Iterations with smaller offset distance inside area C leads to a smaller area around point A.

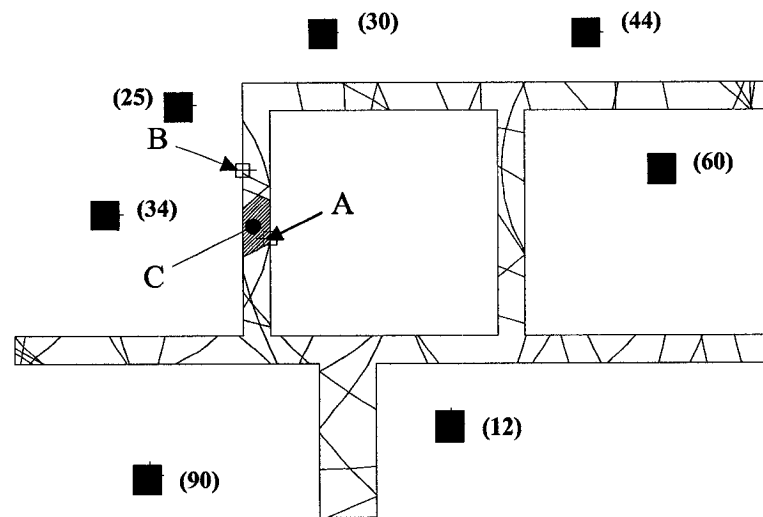


Figure 6.3. Geometric analysis of corridor area for water fountain problem

6.3 CASE EXAMPLE 2: MULTIPLE FACILITIES, SINGLE CONSTRAINT

This example considers a construction site and is obtained, as well, from the literature (Mawdesley et al. 2002) to allow for comparison of results. The site considered in this problem is a $400 \times 200 \text{ m}^2$ rectangle, accommodating six permanent facilities: factory, factory car park, lorry park, office, office car park, and gatehouse. These facilities along with a river and an existing road on the side of the site are modeled as site objects. Figure 6.4 shows the simplified site conditions and location of permanent facilities as considered by Mawdesley et al. (2002).

The goal in this example is to find the optimum location for four temporary facilities: temporary office, reinforcement store, concrete batching plant and general store. It is assumed that each temporary facility occupies a $20 \times 20 \text{ m}^2$ area on site. In the analysis

presented in Mawdesley et al. (2002), the distance is measured in units of 20m for simplicity, and hence each temporary facility occupies one unit on site. Although CASL is capable of considering precise measurements of distances and actual size of objects, the same assumptions are maintained here to enable a comparison.

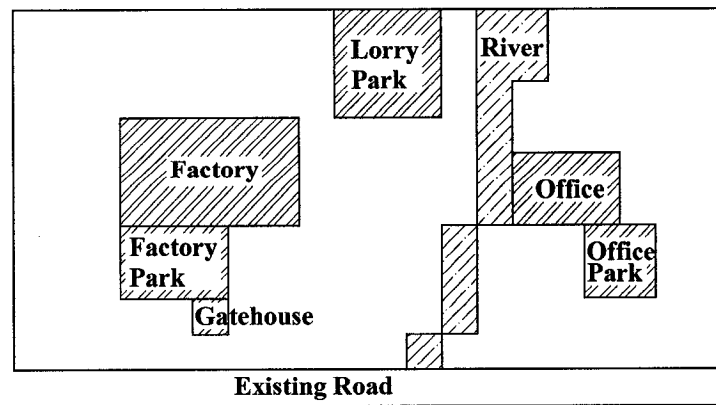


Figure 6.4. Permanent facilities and existing site objects (Mawdesley et al. 2002)

Data related to resource requirements for construction objects (i.e. temporary facilities) is included in Table 6.3. Traffic and transport requirements between two facilities in Mawdesley et al. (2002) are modeled as weights of closeness constraint assigned to construction objects. Mawdesley et al. (2002) consider these requirements from temporary facility A to B different than that of B to A. Since in CASL the weight of a constraint is direction-independent, the closeness weight is generated as the sum of resource requirements from construction objects A to B, and B to A (Table 6.3, see part b). Further, in the work of Mawdesley et al. referred to above, a high set up cost is considered for areas occupied by construction and site objects to avoid the allocation of these areas to new construction objects. To guarantee the non-overlap requirement in CASL, the areas occupied by a located construction or site object is deducted from the

available site area upon their placement on site. The available site area is updated every time a construction object is located on site. Figure 6.5 depicts the available and unavailable areas of the site at the commencement of site layout.

Table 6.3. Closeness weights for four construction objects, adapted from Mawdesley et al. (2002)

		Constrained Elements (Construction Objects)			
		Temporary office	Reinforcement store	General store	Concrete batch plant
Constraining Elements	(a) Site Objects				
	Factory	1200	700	200	150
	Factory Parking	0	0	0	0
	Gatehouse	100	10	50	10
	Lorry park	60	200	0	100
	Office	800	200	500	50
	Office park	20	0	0	0
	Road	20	0	10	0
	(b) Construction Objects				
	Temporary Office	0	100	200	100
	Reinforcement store	100	0	50	0
	General store	200	50	0	100
	Concrete batch plant (CBP)	100	0	100	0
Total weight of constraints		2600	1260	1110	510

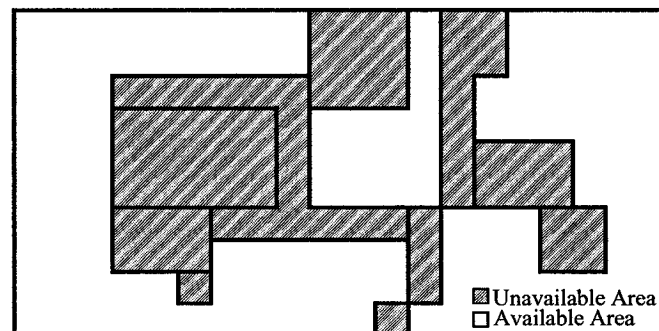


Figure 6.5. Available and unavailable site area

To analyze site space, the closeness rings, with incremental offset distances of 3 units (60 meters), were generated. Figure 6.6 illustrates the analysis for temporary office, entering the site as the first object, with six closeness constraints (see Table 6.3). The site is divided into 177 segments, ranked based on their respective total utility and assigned a color relative to their rank. The segments ranked 1 to 4 are identified in Figure 6.6. To accommodate the assumptions of Mawdesley et al. (2002) and enable a closer comparison of layouts, the final location of objects were matched with grid points. As such, after dividing the site into segments and finding the best segment, the final location of each object was matched to the grid point that falls in that segment. Should more than one grid point exist in a designated segment, the search would be fine-tuned inside that segment using more closely spaced closeness rings.

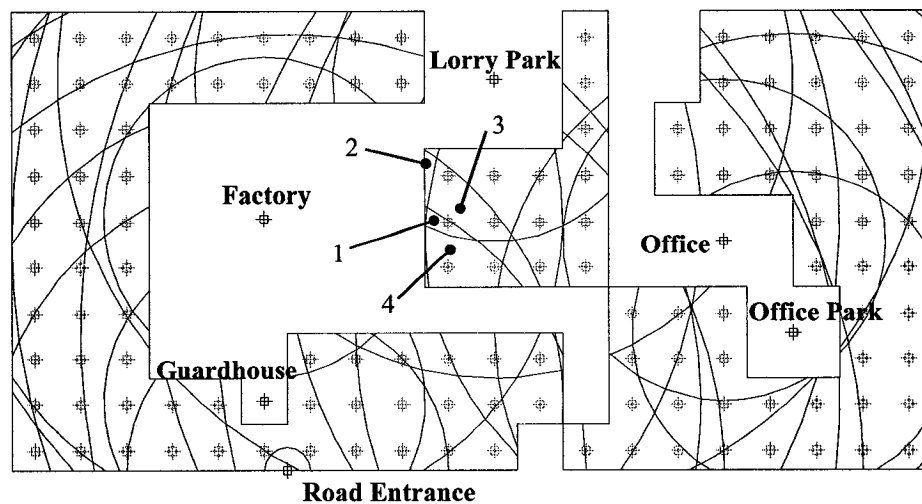


Figure 6.6. Site analysis for temporary office

The developed model is utilized in several trials to produce site layout plans. Figure 6.7 shows four layouts for the site addressed in this case example. The layout presented in Figure 6.7a is the one suggested by Mawdesley et al. (2002) and is brought here for

comparison purpose. Figure 6.7b shows the layout resulted from considering the closeness constraints between construction objects and site objects only, i.e. only C-S constraints, shown in Table 6.3-part (a). Figure 6.7c and 6.7d are layouts produced as a result of considering both C-S (Table 6.3- part a) and C-C (Table 6.3- part b) closeness constraints. When generating layouts 3 and 4, once a construction object is located on site, the closeness weights of the newly located object are considered for locating the succeeding objects. The difference between layouts 3 and 4 is in the order of construction objects in the queue, i.e. the order in which they are entered to the layout. Layout 3 (Figure 6.3c) is generated based on the rule of thumb indicating that the most constrained object (i.e. object with largest total weight value) should be located earliest. Applying this rule, the objects are entered to the site in the order of: 1) temporary office, 2) reinforcement store, 3) general store, and 4) concrete batching plant (see Table 6.3). Layout 4 (Figure 6.3d) was found to maintain the minimum total weighted distance out of the $4! = 24$ possible scenarios resulting from different sequential entering of facilities on site. The queue of construction objects for layout 4 includes: 1) temporary office, 2) reinforcement store, 3) concrete batching plant, and 4) general store.

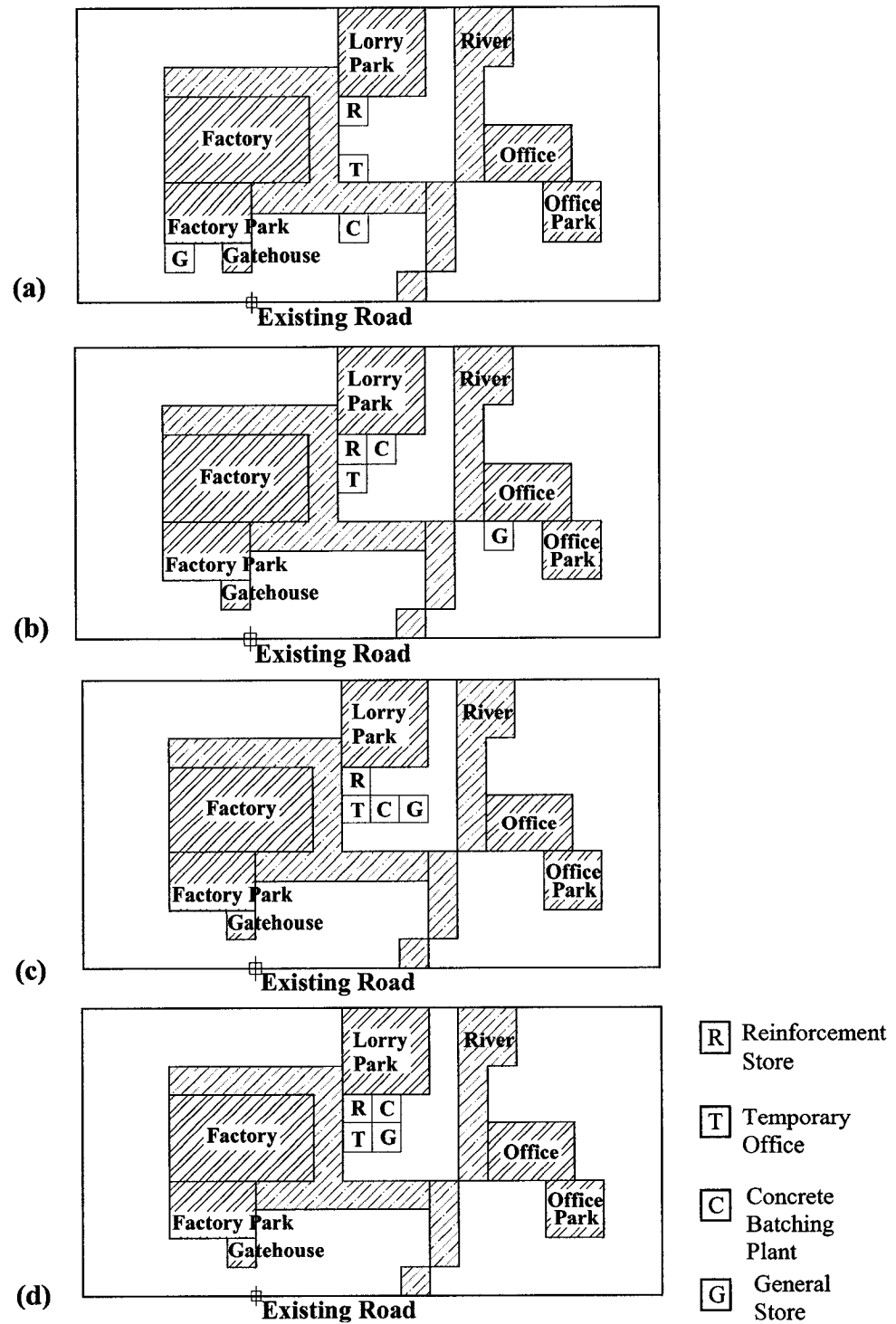


Figure 6.7. (a) Layout 1, suggested by Mawdesley et al. (2002); (b) Layout 2, generated considering the weights of permanent facilities; (c) Layout 3, generated considering weights of temporary and permanent facilities; (d) Layout 4, generated considering weights of temporary and permanent facilities

As it can be inferred from Figure 6.3, layouts 3 and 4 are very similar. The main difference between layout 2 on one hand, and layouts 3 and 4 on the other hand, is in the location of general store. The reinforcement store gets the same positions in all layouts and the relocation distance of temporary office and concrete batching plant is negligible. The location of general store in layout 2 can be explained by the weights assigned to the closeness constraint. Since in layout 2, C-C constraints are not considered, the office is the dominant attraction factor for the general store (see Table 6.3). In layouts 3 and 4 however, the C-C constraints are included in the analysis, and consequently the general store is attracted to a location closer to other construction objects (see Table 6.3 and Figure 6.7c and 6.6d).

Table 6.4 illustrates a comparison between the total utility score (U_t) and its breakdown for the four layouts presented in Figure 6.7. Since the only constraint considered in this example is closeness, Equation 4.2 can be rewritten in terms of total weighted distance in site units (i.e. 20 m). As such:

$$f = \sum_{i=1}^n \sum_{j=1}^m (d - d_{ij}) \cdot W_{ij} \quad 6.3$$

Where d is the longest dimension of the site; d_{ij} is the distance between i th constrained element and j th constraining element; and W_{ij} is the weight of the closeness relationship between the i th and j th objects.

As inferred from Table 6.4, CASL is able to generate site layouts with better total satisfaction score. Layout 2, with considering the closeness of permanent facilities only, holds a larger total weighted distance than layout 1. However, layouts 3 and 4 prove that considering both C-C and C-S constraints generates better general satisfaction of constraints. The difference between the total weighted distance of layouts 3 and 4 is negligible. This shows that the rule of thumb used to determine the order of entering the objects to site in layout 3 can result in very close to optimum solutions.

This problem displays the capability of the proposed model to analyze a site layout problem; nevertheless it does not demonstrate all of its features. The model is capable of modeling more realistic details with different levels of precision. For example it can consider the actual dimensions and shapes of objects, and account for other locating constraints such as the minimum distance desired between two selected objects and safety-related constraints. In comparison to genetic algorithms used by Mawdesley et al. (2002), CASL was able to better satisfy the objective function.

Table 6.4. Total utility score breakdown for layouts in Figure 6.7

Facility Name	Layout 1	Layout 2	Layout 3	Layout 4
Temporary office	39479.62	40020.98	40739.44	40898.02
Reinforcement	19975.13	20174.10	20353.84	20394.93
General store	12585.98	17033.22	17934.29	17925.60
Concrete Batch Plant	7667.21	8262.00	8634.72	8662.00
Total utility score (U_i)	79707.94	85490.30	87662.29	87880.55

6.4 CASE EXAMPLE 3: MULTIPLE FACILITIES, MULTIPLE CONSTRAINTS

This case example is an actual case from the industry, developed to illustrate the capabilities of CASL, and its adaptability with actual construction practice. This project was developed with the aid of the project's site engineer. However, it was not a hands-on experience, since the aim of this exercise was to establish the functionality of the model, and not the convenience of its user interface. The site engineer was given a tutorial that demonstrated how CASL works at the project setup phase and site analysis and allocation phase, through its user interface. The data and knowledge required for setup including the involved site, construction, and constraint objects, and queuing attributes were transferred from the site engineer on paper, using drawings and photographs of the job site. This information was then used in CASL to setup the Project Palette. Based on these settings, the site was analyzed and construction objects were located on site. At different decision-making stages the site engineer was consulted. The final product was presented to site manager for perusal and comments.

Project description: The problem involves designing a layout for the LG-2 project's support facility site, which is a major hydro project constructed in Quebec, Canada. The site has an irregular shape with an approximate area of 28'178 m² and was required to accommodate the thirteen facilities listed in Table 6.5. The site has two entrances on the main road, one considered mainly for cars and the other for equipments. An access road, parallel to the main road, is also considered inside the site for on-site traffic. Figure 6.8 shows the layout of the facilities on site as designed by the project team. The numbers

refer to the construction objects listed in Table 6.5. Appendix III contains photographs depicting the LG-2 project site organization.

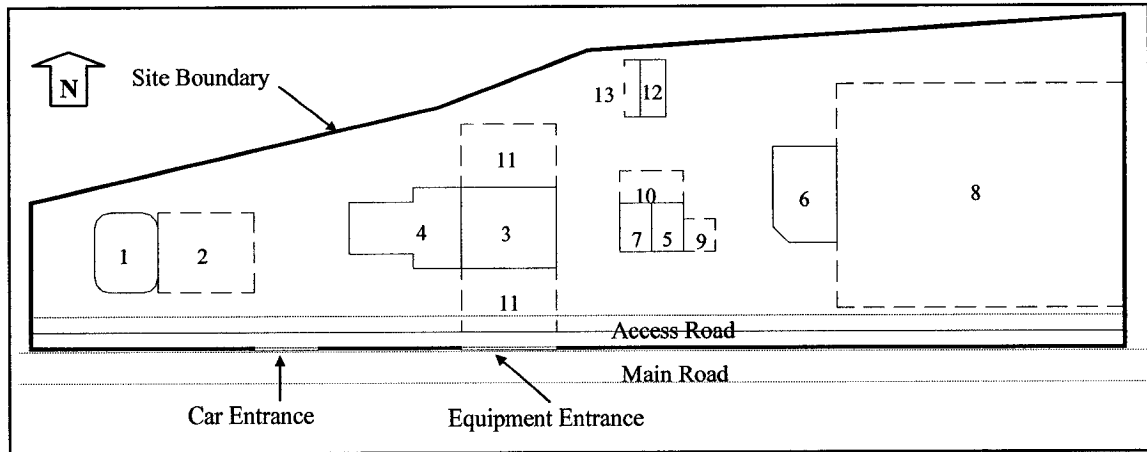


Figure 6.8. Actual site layout of LG-2 project

Table 6.5. Data pertaining queuing of construction objects for LG-2 project

Construction object	Queuing Attributes				Q	Rank
	Area (m ²)	Number of constraints	Weight of constraints	Number of constrained els.		
1 Office	500	10	760	8	12.396830	1
2 Parking	750	2	160	1	2.737343	10
3 Garage	750	6	470	5	7.968922	2
4 Storage	320	5	340	2	5.043442	6
5 Welding shop	150	4	300	4	5.297995	5
6 Plumbing shop	600	3	190	3	4.176190	9
7 Electrical shop	150	4	330	4	5.455890	4
8 Formwork storage area	6300	3	190	1	7.700000	3
9 Steel storage area	100	2	150	2	2.668839	11
10 Electrical storage area	200	1	100	1	1.585046	12
11 Equipment parking	1200	4	260	2	4.920802	7
12 Soil & concrete test labs	144	4	290	3	4.740602	8
13 Lab parking	90	1	100	1	1.497744	13
Max Value (C_{Max})	6300	10	760	8		
Weight (W_c)	5	4	4	4		

Project modeling: Mapping the LG-2 project in CASL, the Project Module starts with generating the required objects. After generating the site boundaries, the three identified site objects (i.e. car entrance, equipment entrance, and access road) along with their respective geometry were generated in Site Module and they were placed on their known location on the site. Each edge of the site boundary was also identified as a site object. As well, three regions on the site, highlighted in Figure 6.9 as A, B, and C, were recognized as “unavailable” by the site engineer. Region A is a space with 10m in width, inside the site boundary, provisioned as a buffer space to keep construction objects from being located on the edge of the site. Regions B and C indicate 20m of clearance around the entrances to the site. These three regions were modeled as site objects to avoid the allocation of their area to other construction objects.

Thirteen facilities were modeled as the construction objects and added to the Construction Palette. For each of the construction objects, a number of constraint objects were defined by the site engineer. Table 6.6 summarizes the constraint objects, their composing elements, and priority weights assigned to each construction object. Once the Project Palette is set, the Queuing Module ranks the order of construction objects. Four queuing attributes, i.e. area, number of constraints, weight of constraints, and number of constraining elements, were selected by the site engineer and assigned a priority weight to create the queuing criteria. The data associated with the selected attributes is summarized in Table 6.5 (columns 3 to 6). It should be noted that this data was retrieved directly from the model’s database, upon forming the queuing criteria. Table 6.5 also contains the calculated queuing score (Q) and the rank of each object in the queue.

Table 6.6. Constraint objects defined for the LG-2 project

No.	Constrained element	Relation	Constraining element	Weight
1	Office	far from	Electrical shop	90
		far from	Welding shop	100
		far from	Plumbing shop	90
		not within 100m of	Garage	50
		not within 5m of	Access road	50
		close to	Car entrance	80
		close to (linear)	West edge of site	100
		next to	Parking	90
		close to	Storage	50
		far from	Steel storage area	60
2	Parking	next to	Office	90
		close to	Car entrance	70
3	Garage	within 50m of	Equipment entrance	80
		not within 100m of	Office	50
		next to	Storage	100
		next to	Equipment parking	100
		south of	Equipment parking	80
		close to	Soil & concrete test labs	60
4	Storage	close to	Office	50
		visible from	Office	70
		next to	Garage	100
		close to	Car entrance	40
		close to	Equipment entrance	80
5	Welding shop	next to	Plumbing shop	40
		far from	Office	100
		not within 80m of	Formwork storage area	60
		next to	Steel storage area	100
6	Plumbing shop	close to	Welding shop	40
		far from	Office	90
		close to	Electrical shop	60
7	Electrical shop	close to	Plumbing shop	60
		far from	Office	90
		next to	Electrical storage	100
		close to	Soil & concrete test labs	80
8	Formwork storage area	not within 80m of	Welding shop	60
		close to (linear)	East edge of site	80
		close to (linear)	Access road	50
9	Steel storage area	next to	Welding shop	100
		far from	Office	60
10	Electrical storage area	next to	Electrical shop	100
11	Equipment parking	next to	Garage	100
		close to	Car entrance	10
		close to	Equipment entrance	70
		north of	Garage	80
12	Soil and concrete test labs	close to	Garage	60
		close to	Electrical shop	80
		close to (linear)	North edge of site	50
		next to	Lab parking	100
13	Lab parking	next to	Soil & concrete test labs	100

Analysis and layout generation: Upon queuing of construction objects, the Spatial Analysis Module is activated to find a location for each object according to the constraints defined for them. The site engineer accepted the queue of objects based on their ranks in Table 6.5 provided two modifications: electrical shop was moved up in the queue to be located after the formwork; and plumbing shop was moved down in the queue to be located after test labs. Such modifications are supported in CASL and as such the queue of objects yields the following order: office, garage, formwork storage, welding shop, plumbing shop, storage, electrical shop, equipment parking, soil and concrete test labs, parking, steel storage, electrical storage, and lab parking. For each construction objects, the site is analyzed considering the constraints presented in Table 6.6. Some of these constraints are general layout rules that can apply to any construction project. Others are project-specific, being defined for the special conditions of the LG-2 site. Following is a description for site analysis process for each of construction object in the order they appear in the queue.

1. Office

The first object to locate is the office. At the commencement of the layout only site object are located on the site. Therefore, three of the defined constraints in Table 6.6, whose constraining elements are site objects, are applicable to find a suitable location for the office (see Table 6.6). These constraints indicate the office to be close to the car entrance, close to the west edge of site, and not within 5 meters of road. Figure 6.9 shows the site analysis, as well as the initial and the final location for the office. The object is located on the segment with the highest satisfaction score for the three aforementioned constraints.

In all the following figures presented for the site analysis, this segment is highlighted as D. The numbers shown in those figures represent the satisfaction score (U) for a set of site segments closer to the segment D. The objects are first placed in the segment D, matching their centroid with that of the found segment, denoted with a point. The lightweight line in these figures represents the initial position of the object located by the model. The location of the office is then fine-tuned based on site conditions and directions from the site engineer, and presented with a thick line. In Figure 6.9, the location of site is modified so as not to overlap with the unavailable area A.

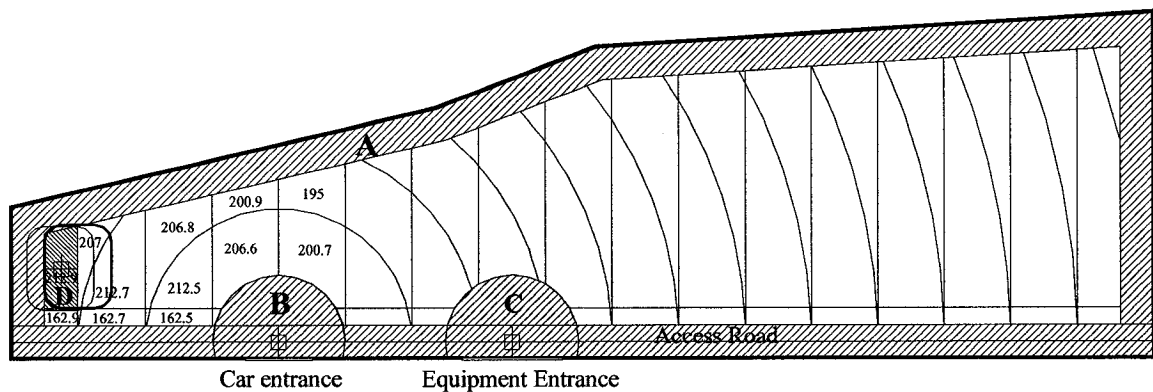


Figure 6.9. Site analysis for office

2. Garage

Garage is the second object in the queue, with two applicable constraints (see Table 6.6). Both constraints are general layout rules of thumb. Since in this project the construction job was located outside the facility layout site, equipments had to commute between the two. To minimize on-site travel distance one constraint indicates that the garage should be located within 50 meters of the equipment entrance. To provide the office a certain distance from the source of noise, the second constraint keeps the garage at least 100 meters away from it. Figure 6.10 shows the initial location of the object in lightweight

line. It is then moved, so as not to overlap with region C, marked unavailable. To grant a general organization to the layout, when modifying the location of objects and selecting the final position, the effort was to align objects with the existing ones on the layout. As such, in its final location, the center of garage is aligned with that of equipment entrance.

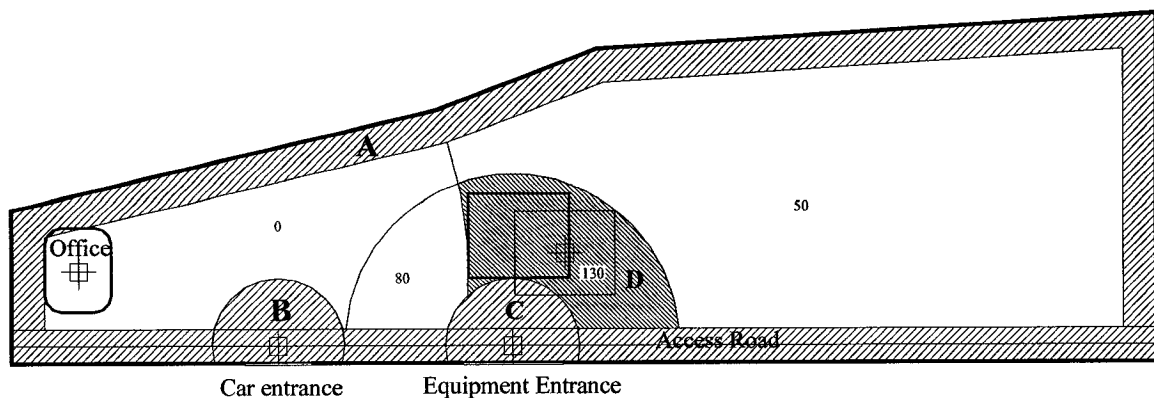


Figure 6.10. Site analysis for garage

3. Formwork storage area

Out of the three constraints defined for the formwork storage area, two were applicable at this point: linear closeness to the access road and to the east edge of site. The latter constraint was assigned to keep the storage area for formwork towards the east side of the site area due to its remoteness from the entrances, where most of the traffic was expected to happen. These two linear constraints divided the site area into rectangular segments as shown in Figure 6.11. Evidently, the segment in the bottom-right corner gets the highest score. Refining the location of the object, it was first moved inside the available site area, invoking the corner-to-corner adjacency. The footprint of the object at this location is marked with the dashed line. Then it was decided to change the shape of the formwork storage to benefit from the unused area to its north, where other objects cannot be

located, and free the space to its west. Since this object was *flexible*, resizing was permitted. The thick line in Figure 6.11 marks the final location of the storage area for the formwork.

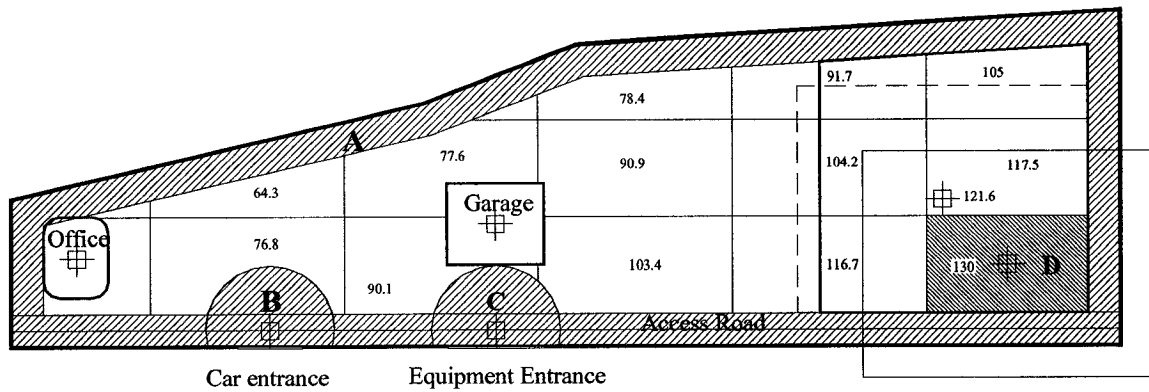


Figure 6.11. Site analysis for formwork storage area

4. Welding shop

Considering the current status of site, two of the constraints assigned to welding shop were applicable, indicating its location to be far from the office and not within 80 meters of formwork storage. Both of these constraints are general rules and can be applied to other projects. Generally, it is preferred to located workshops far from the office to keep the noise away. As well, for safety reasons, welding shops should have a certain distance from the formwork storage. The highlighted area D in Figure 6.12 best satisfies both constraints. The model-assigned location for welding shop is shifted south, to align the object with the garage in the middle.

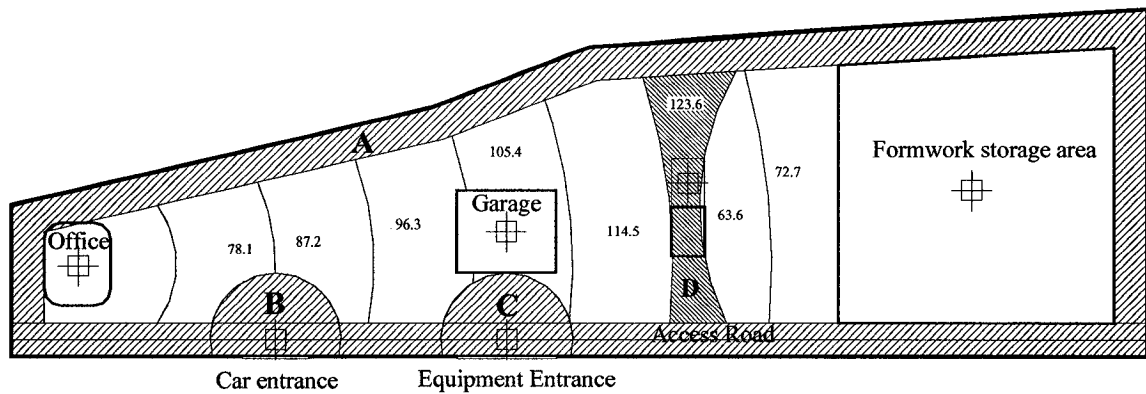


Figure 6.12. Site analysis for welding shop

5. Plumbing shop

The two applicable constraints defined for plumbing shop include being far from the office and close to the welding shop. These two, as well, are layout rules of thumb. As mentioned for the previous object, to keep the noise level down for the office, workshops are generally located away from it. As well, as a general rule of thumb, workshops are preferred to be located close to one another due to their similarity in activities. Specifically when fabrication of the pipes is preformed onsite, such as in this project, the plumbing shop should have a fair closeness to welding shop. The final location is fine tuned, first to be corner-to-corner adjacent to the formwork storage area (shown with dashed line in Figure 6.13), and then aligned with the garage in the bottom.

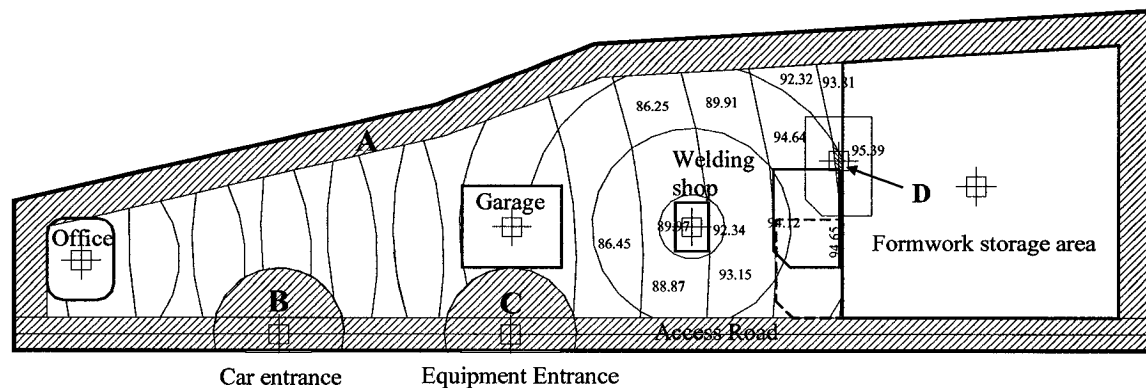


Figure 6.13. Site analysis for plumbing workshop

6. Storage

Figure 6.14 shows the site analysis for the storage as the sixth construction object to be laid on site (see Table 6.5), considering five constraint objects (see Table 6.6). These constraints indicate that the storage should be located close to the two entrances to ease deliveries; close to the office since several visits to the storage were predicted, and visible from office to maintain security. The constraint object specifying the adjacency between the storage and garage was specific to this project since it was intended to use the same structure for the two. When the available site area is analyzed based on the constraints, the segment highlighted as D receives the highest satisfaction score and accordingly the object is first placed at the location. The location of storage is further fine-tuned in the locating module and it is aligned with the garage, invoking the corner-to-corner adjacency feature.

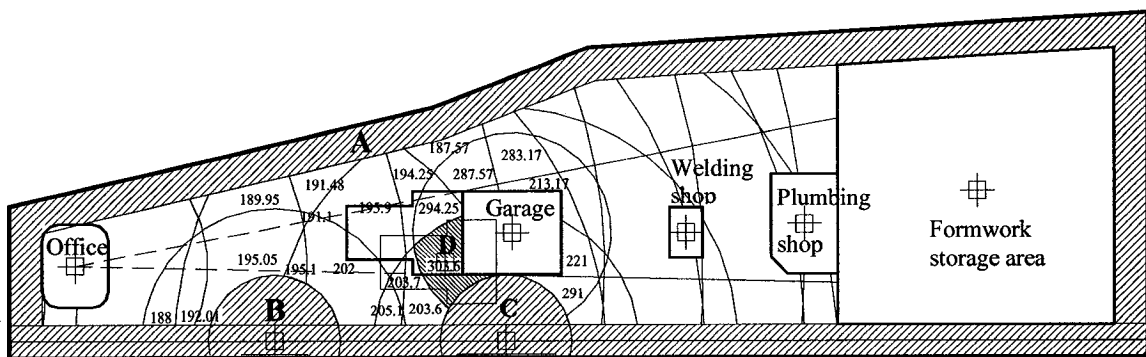


Figure 6.14. Site analysis for storage

7. Electrical shop

Two general layout rules on locating workshops, discussed previously, were used to locate this object. It was required to keep it far from office, and close to plumbing shop, in accordance with the aforementioned rules of thumb for workshops. Figure 6.15

illustrates the site analysis for the electrical shop. After the initial locating of object on the found location, its location was adjusted to be aligned with the corner of plumbing shop.

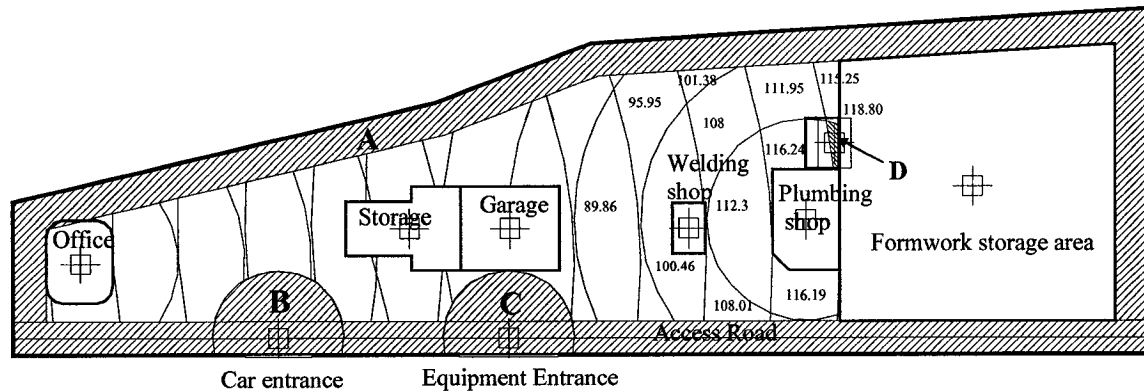


Figure 6.15. Site analysis for electrical shop

8. Equipment parking

This object was constrained to be next to the garage, close to equipment entrance, loosely closed to car entrance, and on the north side of garage. The latter was a preference of site engineer and hence, project specific; while the first three can be applied to any construction project. The lightweight boundary of parking equipment in Figure 6.16 represents its initial location on site. Since the object was overlapping with the buffer area A, it was decided to change its shape aligned with the edge of buffer area. Since the parking was *flexible*, the model allowed the reshaping of this object. Figure 6.16 demonstrates the final shape and location for the equipment parking.

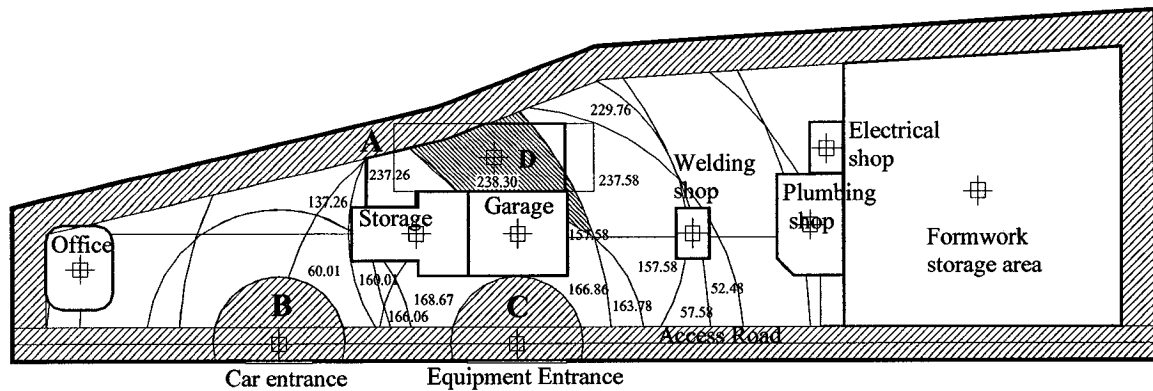


Figure 6.16. Site analysis for equipment parking

9. Soil and concrete test labs

This object was constrained to be close to garage and electrical shop, due to the cooperation predicted, and hence the travel effort. As well, as a project specific constraint, it was constrained to be closer to the north edge of the site (see Table 6.6). When the segment that best satisfies the three aforementioned constraints was found, and the object was located in its initial position, it was decided to rotate the test lab so that its entrance faces both garage and electrical shop. Figure 6.17 illustrates the initial and final location of the soil and concrete test labs.

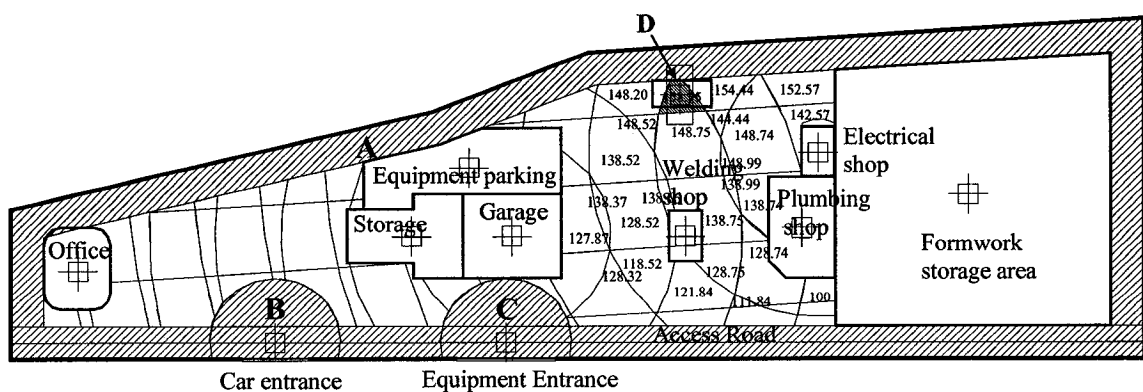


Figure 6.17. Site analysis for soil and concrete test labs

10. Parking

Two layout rules of thumb constrained the location of the office parking: its adjacency to the office, and closeness to the car entrance. It was located on site by matching its centroid with that of the segment with the highest satisfaction score. Its location was then fine-tuned to have corner-to-corner adjacency with the office, as shown in Figure 6.18.

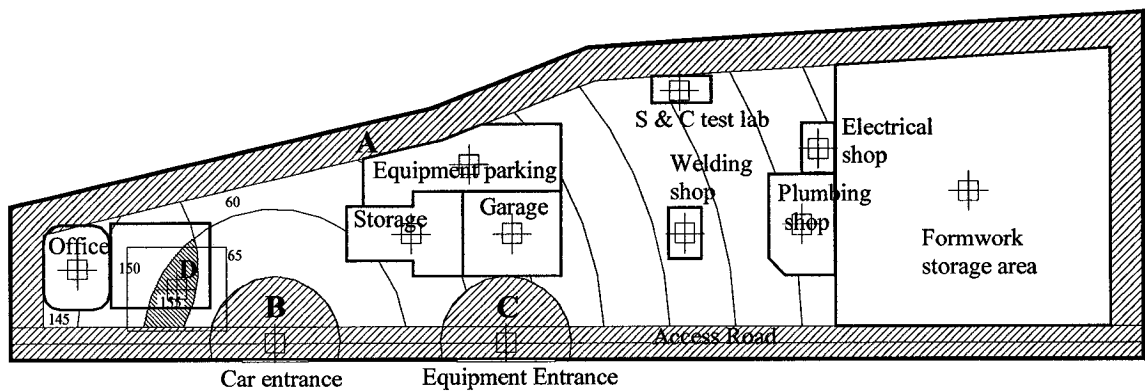


Figure 6.18. Site analysis for parking

11. Steel storage

The steel storage was constrained to be next to the welding shop, to decrease the travel effort between the two, and far from the office. As shown in Figure 6.19, basically the adjacency constraint defines the location of steel storage inside the circle circumscribing welding shop. The farness to the office then delimits its location to the east of welding shop. To refine the location of object, it was then shifted to left, so as not to overlap with the welding shop.

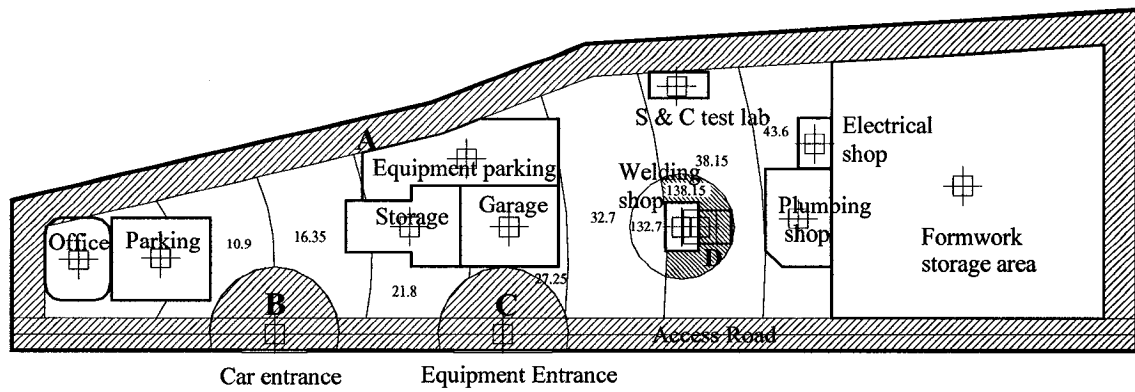


Figure 6.19. Site analysis for steel storage

12. Electrical storage

Only one constraint was defined for this object, that delimits its location next to the electrical shop, as shown in Figure 6.20. The corner-to-corner adjacency feature is invoked to rectify the overlap situation.

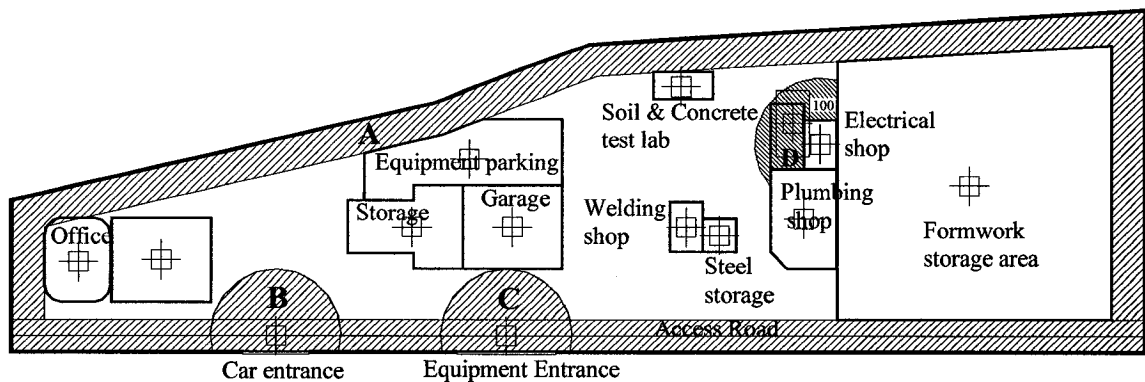


Figure 6.20. Site analysis for electrical storage

13. Lab parking

This object, as well, was delimited by only one constraint: adjacency to the test lab. As shown in Figure 6.21, the object was rotated after initial positioning on site, to be laid aligned with the test lab.

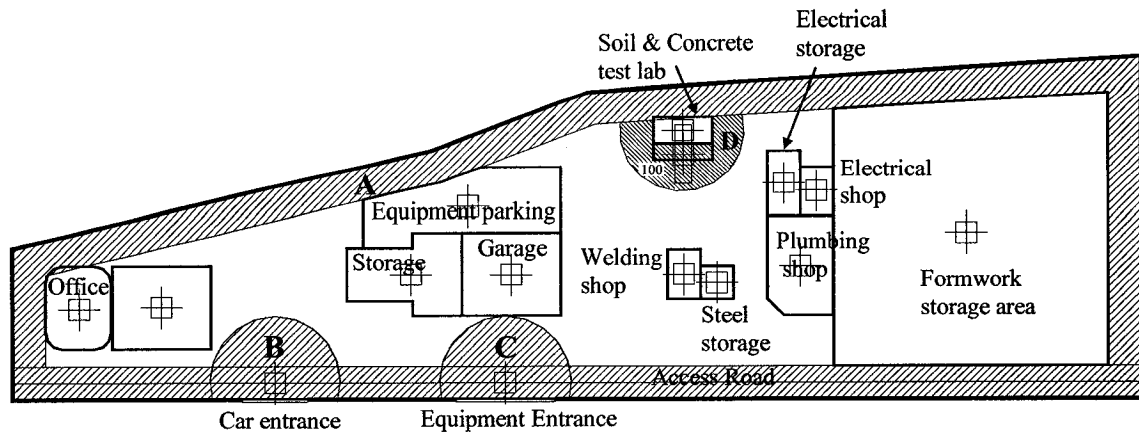


Figure 6.21. Site analysis for lab parking

Discussion: Figure 6.22 shows the layout of the site after the analysis is performed for all the construction objects in the queue. The numbers refer to the construction objects listed in Table 6.5. It is interesting to note the similarities and dissimilarities between the generated layout and the original one. Equipment Parking (11) is moved to one side of the Garage (3) to maintain to clearance area defined for the Equipment entrance. Also its shape was redefined in accordance to the clearance defined for the site boundary, while maintaining the same area. Similarly, the shape of the Formwork Storage Area (8) was slightly resized to take advantage of the unused corner of site and give space on its left, which is in continuity with the open space. Electrical Shop (7) in the proposed model is shifted to the right of the site to better accommodate the two constraints of closeness to the Plumbing Shop (6) and farness from the Office (1). In general, the model was able to generate a layout that is close to the original site arrangement with some differences that were in favor of the defined constraint objects. Using the evaluating module, the generated layout received a total utility score of 3195.3, compared to 3080.49, of the original layout (see Figure 6.8). Hypothetically, an ideal layout that fully satisfies all the constraints gets a score of 3650 (i.e. sum of all the weights assigned to constraints).

Although it is recognized that 100% satisfaction for each and every object may represent a practical impossibility, the “ideal score” can still be used as an attribute to evaluate the efficiency of the various generated layouts. Using this indicator suggests that CASL was capable to create a layout that satisfies the defined constraints to a comparable degree with those generated by the practitioners (87% satisfaction for CASL versus 84% for the original layout).

In order to evaluate the developed model and get a feedback on its essential functions, site engineer was presented with the final layout generated by CASL. He considered the layout satisfactory, and on close examination it became clear that the differences from the actual layout were attributed with the stated objectives and defined constraints. He considered CASL a useful strategic site layout tool, and found its visualization helpful and informative. The site engineer recognized CASL as a tool that can add value to the planning process by identification and allocation of spaces and in generation of a number of what-if scenarios to assist site planners in performing their task. Further, he found it useful as a communication and visualization tool to describe the layout to other parties involved in the project. The site engineer expressed that he could see CASL fitting into the planning procedure and identified the potential benefit in adopting CASL in construction practice.

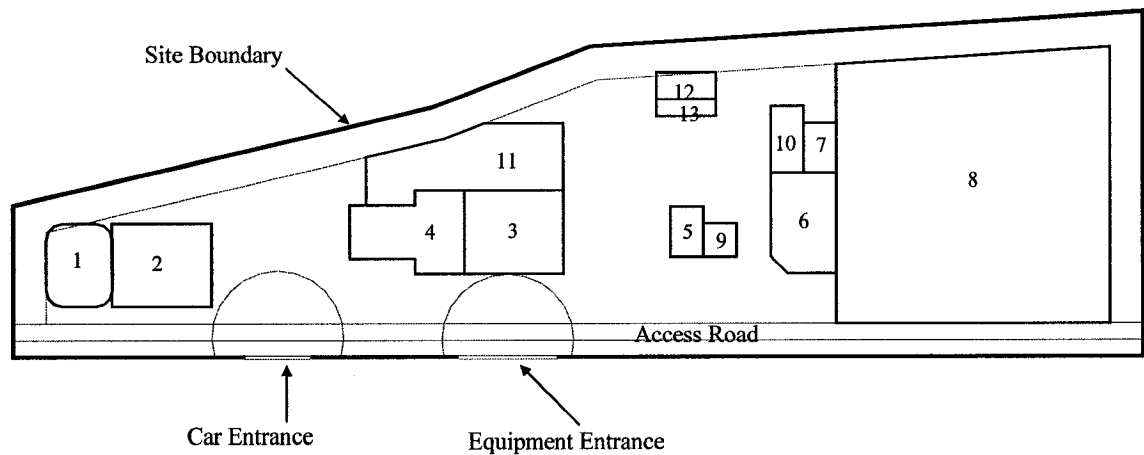


Figure 6.22. Site arrangement for LG-2 generated using the proposed

6.4 SUMMARY

The aim of this chapter was to establish the functionality and usability of CASL. Three numerical examples of site layout, two drawn from the literature and one actual case from industry, were analyzed using the developed model. The first example is a simple case of *location allocation*. The problem is defined as finding the optimum location for a water fountain in the corridor area of a manufacturing office complex, so that it bears the minimum total distance from the offices housed in that complex. When comparing the results with those reported in the literature, the proposed model was able to find similar locations for the water fountain, respecting the original assumptions. The second case example is a *construction* site layout problem, also drawn from the literature. It involves locating multiple facilities on site, considering closeness relationships among them. The results indicate that CASL was able to generate a layout that satisfies the optimization objectives better than the model described in the literature.

These two examples illustrate the functionality of CASL and its accuracy, in comparison to models developed by others, in finding near-optimum locations for construction objects. However, the objective considered in the two examples is limited to minimizing travel distance, and as such they do not fully demonstrate the capabilities of CASL to conduct spatial search based on multiple constraints. Unlike the previously developed models, CASL can readily accommodate further changes in the setting of a project and account for other locating constraints. To further demonstrate the capabilities of CASL, a case example of an actual project was developed and analyzed with the assistance of the project's site engineer. This problem involved locating of multiple facilities, considering a variety of constraints. The results were presented to the site engineer for examination and comments. The site engineer recognized that the generated layout is reasonable and better satisfies the defined constraints, compared to the original layout. As well, he considered CASL to be a useful site layout tool, and found the visualization of the model helpful and informative. More importantly, he expressed the potentials of CASL to be incorporated in the planning phase of construction, in identification and allocation of site spaces, and in visualization of construction sites as a mean of describing the site layout to other involved parties.

CHAPTER 7

SUMMARY AND CONCLUDING REMARKS

7.1 SUMMARY

This thesis described site layout problem and highlighted its impact on successful project delivery. A proper site layout enhances productivity on job sites by minimizing travel time on sites, decreasing material handling effort, increasing safety, and hence decreasing the total cost of project. Despite its importance, it is often neglected in planning, engineering, and construction phases of a project. This can be associated with the ill-structured nature of required information, which renders site layout process knowledge intensive, time consuming, and hence, expensive.

This thesis presented a review of the existing site layout models focusing on two aspects: 1) problem representation, and 2) problem solving approach. It was recognized that existing models suffer from lack of a formalized structure for problem representation. Further, the problem solving approach used in these models does not represent the way site planners approach the layout.

Devising a formalized structure to represent site layout, the elements that affect site layout were identified and classified into three categories of site objects, construction objects and constraint objects. An object-based approach was utilized to represent the three tiers of objects. For each tier of objects, the characteristic attributes required for the

site layout procedure were identified. Accordingly, each tier is represented as an object with a set of attributes which are categorized into three groups of geometric, graphic, and layout (non-geometric). Defining a formalized structure for objects is specifically important for ensuring the flexibility of the model since it facilitates the creation of new ones. To create new objects in each tier, it is sufficient to invoke object's structure and assign values to its attributes.

To aid planners with finding optimum or near optimum location for construction objects on site, a geometric reasoning approach for analyzing site space was developed. This approach implements a visual site layout process that is comprehensible to planners. The developed space analysis methodology is based on geometric representation of constraint satisfaction. The constraint satisfaction scheme, in turn, is a function of the spatial relationship used in the structure of the constraints in this research. Six groups of possible spatial relationships among objects were identified. A geometric representation for each relationship, along with its satisfaction scheme was introduced. Analyzing site space in search of the optimum location for a construction object is performed through a visual constraint satisfaction scheme. The overlapped representation of all the constraints assigned to an object generates an overall constraint satisfaction map on site boundary. This map identifies areas on site that best satisfy the considered constraints and presents the level of constraint satisfaction for the rest of the site space. This gives the planner an overall view of the site, and aids in determining the final location for a construction object.

Based on the developed problem representation and space analysis approach, a framework for a CAD-based site layout model was developed. The structure of the model is comprised of three main components: 1) database, 2) Project Module, and 3) Layout Control Module. The database is consisted of two sections: Model Library (ML), and Project Palette (PP). Model Library acts as an object gallery and stores objects of three tiers in their respective libraries; namely site library, construction library, and constraint library. Project Palette stores a record of problem-specific objects. Project Module assists planners in the setup phase of a project. Required objects from three tiers are *defined* in this module. The term *define* denotes selection from the respective libraries in ML, or creating new ones. Newly defined objects will then be added to their respective libraries in ML for future reference. As such, the model adopts an open architecture to accommodate projects with different settings and requirements. Therefore, unlike other models, it is not limited to one project setting. This allows the planners to apply their individual problem solving approach and contribute directly to the knowledge and information stored in Model Library. As well, the model gets adapted to the planning strategy of individual planners through the course of time.

Since the layout method adopted in this research is construction method, the generated layout is path-dependent; i.e. it depends on the order in which construction objects are entered to the layout. Therefore, a method that allows site planners to define multi-attribute criteria for queuing objects was developed. Using this feature, site planners can define the priority among construction objects based on their knowledge and uniqueness

of each project. Once formed, the queue of construction objects is sent to Layout Control Module to place them on the layout.

The Layout Control Module is comprised of three sub-modules: Spatial Analysis, Locating, and Evaluating Modules. Spatial Analysis Module utilizes the developed geometric analysis methodology to find the fittest location for each construction object based on the assigned constraints. Locating Module aids the planner in finalizing the location of the construction objects and places them on site boundary. It also keeps track of the available site space when locating construction objects. Thus, the model provides a feedback on whether the space is sufficient to accommodate objects. Based on the locating constraints defined for the project, Evaluating Module measures a total utility score for the generated layout to find its overall fitness as compared to others layouts. Different layouts can be generated by changing the order of objects in the queue.

CASL (Computer Aided Site Layout), a prototype of the proposed model, was implemented using Visual Basic for Applications (VBA) in AutoCAD® environment and employing Microsoft Access® as model's database. The geometries of objects are generated in AutoCAD, where the geometric and graphic properties of objects can be accessed from the built-in database of AutoCAD by VBA via ActiveX technology. Through a unique identification property, these properties are coupled with the non-geometric ones stored in Access. This provides a unique integration between the two databases, in which the non-geometric properties stored in Access can be accessed and

modified from within AutoCAD. As such, VBA provides a seamless integration between AutoCAD and Access, supported by a powerful Graphical User Interface (GUI).

Three case studies were used to validate the viability of the developed site layout model. The first two cases involve locating limited number of construction objects and one type of constraint. These examples are drawn from literature and are aimed to demonstrate the functionality and accuracy of CASL compared to other models. The developed model was able to generate similar or better layouts in both cases. The third case example is a site layout developed from an actual project in Quebec, Canada. This example involved locating a variety of irregular-shaped construction objects on the jobsite, considering different locating constraints. The model was able to generate a layout that better satisfied the defined objective function than that created by site planners. This conclusion was, as well, approved by the site engineer of the project.

This tool assists site planners in efficiently carrying out various aspects involved in site layout planning. Since site layout is mainly based on expert's judgment, special attention was given in design of CASL to allow for user interaction and intervention throughout the process. This feature allows planners to implement their individual approach based on their experience and knowledge. The flexibility of the model in site representation makes CASL compatible with the nature of actual practice in construction sites. The graphical space analysis methodology aids to grant an actual perception of the procedure to site planners, leading their interaction and intervention in the right and effective direction. This in turn, aids in generation of site layouts that better satisfy the defined objectives.

7.2 CONTRIBUTIONS

The contributions of this research are grouped into the following three areas:

1. Site layout representation

- Mapping the site environment into an object-based representation consisting of three tiers of objects.
- Formalizing the structure of objects by identifying their characteristic site layout-related attributes.
- Detecting and structuring a common grammar for expressing spatial constraints among temporary and permanent facilities on site.

2. Site space analysis

- Developing an optimization methodology for locating temporary facilities on job sites using a geometry-based reasoning approach to analyze site space.
- Defining a structure for multi-attributed queuing criteria to assist site planners articulate the order in which objects are entered the layout.

3. Site layout model design

- Designing a framework for construction site layout modeling that assists site planners in generating layouts. This design includes the architecture of its modules, their functionality, and interconnectivity.

- Introducing an open architecture concept for the model to achieve flexibility to accommodate varying site layout projects setups.
- Implementing CASL, a computer-aided site layout tool based on the developed model in an integrated AutoCAD-VBA-Access environment.
- Developing case studies to demonstrated the functionality and evaluate the accuracy of the developed model.

7.3 LIMITATIONS OF THE MODEL

- CASL utilizes a 2D representation for construction site and temporary facilities. Although the height of the facilities is considered in some of its features such as determining visibility between facilities, it does not consider a 3D simulation of site and facilities.
- The actual path taken between two facilities is simplified to the direct distance between them, measured in either Euclidean or rectilinear distances.
- Site layout is represented as a static model; as a result it does not reflect the changes that occur on construction sites through the course of time, automatically. To represent these changes at the current status of model development, different layouts should be generated for different time intervals. For each time interval, the construction objects located in the previous time interval are modeled as site objects with known locations.

7.4 FUTURE WORK

Research in the following four areas is recommended for future work:

1. Expansion of scope (*3D modeling*)

- Expanding the modeling aspects of CASL, in object representation and site space analysis, to consider the third dimension. At the current status of CASL, it is possible to generate and store three-dimensional objects. However, the participation of 3D objects in the space analysis needs to be investigated. To enable three-dimensional analysis, the site needs to be represented as a 3D surface (e.g using *3D Mesh* object of AutoCAD), in which the topography of site can be modeled and analyzed.
- Exploring the potential expansion of CASL's applications to include space scheduling.

2. Incorporation of time factor (*4D modeling*)

- Generating dynamic layouts that account for the changes on construction sites through the course of time. The *time of arrival* and *time of departure* attributes of construction and site objects can be used in such modeling.
- Integrating CASL with scheduling tools to automate the temporal data inquiry by associating each construction object with activities that requires the object.
- Expanding the existing space availability feedback of CASL to *time-space* conflict feedback, when incorporating the time dimension to it. Once identified,

the conflict can be rectified at the early planning phase (i.e. either resolve the conflict, or modify the construction schedule).

3. Further development

- Acquiring domain-specific knowledge regarding the selection of facilities and locating rules, through structured interviews and reported studies. This knowledge can be used to develop knowledge-based tools to automatically 1) select/recommend the required facilities based on the scheduled activities; and 2) assign/recommend locating constraints for selected facilities. At the current status, CASL has a feature that retrieves all the constraints assigned to a facility from the database upon the selection of that facility.
- Integrating CASL with remote sensing techniques such as Radio Frequency Identification (RFID) to provide a real time tracking system of objects on construction sites. Objects in the model will be connected to actual facilities on construction sites and their movement or changes of status can be visualized and traced on computer screen. As well, a representation of actual site layout can be automatically generated by identifying the position of objects on site. The differences between *actual* and *planned* layouts can be identified and acted upon accordingly (e.g. rectify the location of facilities on site, or update the planned site layout).

4. Improvement of the existing model

- Considering actual paths taken between objects on construction sites instead of direct distance between them. This involves the identification and incorporation of the on-site paths. GIS can be utilized to identify the shortest among the existing paths.
- Performing a thorough investigation on identifying more spatial relationships, and on the possibility of adding non-spatial ones for defining constraints among facilities.
- Considering the relocation of facilities on construction sites through the course of construction. The acceptability of this relocation should be calculated if this relocation is acceptable considering the *cost of relocation*. As well, the *time of relocation* and its effect on construction schedule should be considered.
- Testing the model on several projects to attain guidelines for start points for user-defined features, to increase the efficiency of layout procedure (e.g. initial offset distance for linear relationships based on object-to-site size ratio and number of constraints; initial weights assigned to queuing attributes based on type of projects and selected construction objects; or threshold value for total utility score of the generated layout based on weight of all the constraints defined in a project).
- Implementing the designed model in an object oriented programming language to upgrade its efficiency and strength for conventional use.

REFERENCES

1. Aalami, F., Haddad, Z., and Fischer, M. (1997) "Improving Project Control by Automating Detailed Scheduling", Proceedings of the 5th Construction Congress, ASCE, Minneapolis, MN, 430-437.
2. Akinci, B., and Fischer (1998) "Time-Space Conflict Analysis Based in 4D Production Models" Proceedings of the international Computing Congress, ASCE, Boston, MA, 342-353,.
3. Akinci, B., and Fischer (2000) "Four-Dimentional WorkPlanner – A Prototype System for Automated Generation of Construction Spaces and Analysis of Time-Space Conflicts", Computing in Civil and Building Engrg., R. Fruchter et al. eds. ASCE, Reston, VA, 740-747.
4. Akinci, B., Fischer, M., Levitt, R., and Carlson, R. (2002) "Formalization and Automation of Time-Space Conflict Analysis", J. of Computing in Civil Engrg., ASCE, 16(2), 124-134.
5. ASP Construction (2001) "Safety Code for the Construction Industry (S-2.1, r.6)", Association Paritaire pour la Sante et la Se curite du Travail du Secteur de la Construction, Anjou, QC.
6. Atzeni, P., and De Antonellis, V. (1993) "Relational Database Theory", Benjamin/Cummings Pubs., Redwood City, California.
7. Autodesk (1999) "AutoCAD Customization Guide", Autodesk Inc., San Rafael, CA.

8. Balkany, C.A., Birmingham, W.P., and Tommelein, I.D. (1991). "A knowledge-level analysis of several design tools." *Proc. Int. Join Conf. on Artificial Intelligence in Designs, AID91*, Butterworth-Heinemann, Guildford, Surrey, UK.
9. Brans, J.P., Vincke, Ph., and Mareschal, B (1986). "How to select and how to rank projects: the PROMETHEE method", *European Journal of Operational Research*, 24 (2), Elsevier Pubs., North Holland, 228-238.
10. Casakin, H. and Goldschmidt, G. (1999) "Expertise and Use of Visual Analogy: Implications for Design Education", *Design Studies*, (20), Elsevier Science, 153-175.
11. Choi, B., and Flemming, U. (1996) "Adaptation of a Layout Design System to a New Domain: Construction Site Layouts" *Proc. 3th Congress held in Conj. with A/E/C Systems*, Anaheim, CA, 718-725.
12. Cheng, M.Y. (1992) "Automated Site Layout of Temporary Construction Facilities Using Geographic Information System (GIS)", Ph.D. Dissertation, University of Texas, Austin.
13. Cheng, M.Y., and Yang, S.C. (2001) "GIS-Based Cost Estimates Integrated with Material Layout Planning", *J. of Construction Engrg. and Management*, ASCE, 127(4), 291-299.
14. Cheng, M.Y., and O'Connor, J.T. (1993) "Site Layout of Construction Temporary Facilities Using Enhanced-Geographic Information System (GIS)", *Proc. of 10th Inter. Symp. on Automation and Robotics in Construction, ISARC*, Houston, TX, 399-406.
15. Cheng, M.Y., Varghese, K., and O'Connor, J.T. (1992) "Management of Spatial Information for Construction Planning and Design Using Geographic Information

- System (GIS)", Proc. of 9th Inter. Symp. on Automation and Robotics in Construction, ISARC, Tokyo, Japan, 393-402.
16. Choi, B., Flemming, U. (1996) "Adaptation of a Layout Design System to a New Domain: Construction Site Layout", Proc. of the 3rd Computing in Civil Engrg. Congress in conjunction with A/E/C systems, ASCE, Anaheim, CA, 711-717.
 17. Chinowsky, P. (1991) "Knowledge-Based Paradigms in Layout Generation", Proc. of Construction Congress '91, ASCE, Cambridge, MA. 369-374.
 18. Collier, E. and Fischer, M. (1996) "Visual-Based 4D Modeling on the San Mateo County Health Center" Proceedings of the 3rd Congresson Computing in Civil Engrg., Anaheim, CA, 800-805.
 19. Davis, M., (1993) "The Engineering Applications of an Integrated GIS and CAD System", Proc. 5th International Conference Computing in Civil Engineering, ASCE, Anaheim, CA, 734-741.
 20. Dawood, N, Sriprasert, E., Mallasi, Z., and Scott, D. (2003) "An Industrial Evaluation of the Virtual Construction Site (VIRCON) Tools" Proceedings of the CIB W78's 20th International Conference on Information Technology for Construction, Waiheke Island, New Zealand, 97-104.
 21. Eastman, C.M. (1999) "Building Product Models: Computer Environments Supporting Design and Construction" CRC Press, Boca Raton, Florida.
 22. Elbeltagi, E., and Hegazy, T.M. (1999) "Genetic Optimization of Site Layout Planning" Proc. of Annual Meeting of International Transactions, AACE, Denver, Colorado, IT.05.1- IT.05.8.

23. Elbeltagi, E., and Hegazy, T.M. (2003) "Optimum Layout Planning for Irregular Construction Sites" Proc. of 5th Construction Specialty Conference, CSCE, Moncton, NB, COG 197.1 – 197.10.
24. Elbeltagi, E., Hegazy, T.M., Hosny, A.H., and Eldosouky, A. (2001) "Schedule-dependent evolution of Site Layout Planning" J. of Construction Management and Economics, Spon Press, 199(7), 689-697.
25. Elmasri, R., and Navathe, S. B. (2000) "Fundamentals of database systems" 3rd ed., Addison-Wesley.
26. Fischer, M., and Aalami, F. (1996) "Scheduling with Computer-Interpretable Construction Method Models", J. of Construction Engrg. and management, ASCE, 122(42), 337-346.
27. Flemming, U., and Chien, S.F. (1995) "Schematic Layout Design in SEED Environment" J. of Architectural Engrg., ASCE, 1(4), 162-169.
28. Flemming, U., and Woodbury, R. (1995) "Software Environment to Support Early Phases in Building Design (SEED): Overview" J. of Architectural Engrg., ASCE, 1(4), 147-152.
29. Foulds, L.R. (1983). "Techniques for facilities layout: deciding which pairs of activities should be adjacent" Management Science, 29(12), 1414-1426.
30. Foulds, L.R., Gibbons, G.B., Giffin, J.W. (1985). "Facilities layout adjacency determination: an experiment comparison of three graph theoretic heuristics" Operations Research, ORSA, 33(5), 1091-1106.
31. Froese, T., (2002), "Connecting the Information Chain", Canadian Civil Engineer, CSCE, 19(2), 15-16.

32. Halpin, D.W., and Woodhead, R.W. (1998) "Construction Management", John Wiley and Sons, New York, NY.
33. Hamiani, A. (1987) "CONSISTE: A Knowledge-Based Expert System Framework for Construction Site Layout", Ph.D. Dissertation, University of Texas, Austin.
34. Hamiani, A. (1989) "Knowledge Representation for the Site Layout Problem" Proc. of the 6th ASCE Conference, Computing in Civil Engrg. , Atlanta, GA, 283-289.
35. Hamiani, A., and Popescu, C. (1988) "Consiste: A Knowledge-Based Expert System for Site Layout" Proc. of Computing in Civil Engrg. 5th Conference (Microcomputers to Supercomputers), ASCE, Alexandria, Virginia, 248-256.
36. Handa, V., and Lang, B. (1988) "Construction Site Planning" Construction Canada, 30(3), 43-49.
37. Handa, V., and Lang, B. (1989) "Construction Site Efficiency" Construction Canada, 31(1), 40-48.
38. Harmanani, H., Zouein, P., and Hajar, A. (2000) "An Evolutionary Algorithm for Solving the Geometrically Constrained Site Layout Problem" Proc. of the 8th ASCE International Conference on Computing in Civil Engrg., Stanford, CA, 1442-1449.
39. Hayes-Roth, B. (1985) "A Blackboard Architecture for Control" Artificial Intelligence, 26, 251-321.
40. Hegazy, T.M., and Elbeltagi, E. (1999) "EvoSite: Evolution-Based Model for Site Layout Planning" J. of Computing in Civil Engrg., 13(3), 198-206.

41. Hegazy, T.M., and Elbeltagi, E. (2000) "Simplified Spreadsheet Solutions: A Model for Site Layout Planning" J. of Cost Engrg., 42(1), 24-30.
42. Holtzblatt, K., and Jones, S. (1993) "Contextual Inquiry: A Participatory Technique for System Design" Participatory Design, Schuler, D., and Namioka A., Erlbaum, Eds., 117-210.
43. IAI (2004), International Alliance for Interoperability, International Homepage at <http://www.iai.international.org>, Accessed July 2004.
44. Jeljeli, M.N., Russell, J.S., Guy Meyer, H.W., and Vonderohe, A.P. (1993) "Potential Applications of Geographic Information Systems to Construction Industry" J. of Construction Engrg. and Management, ASCE, 119(1), 72-86.
45. Jones, C. (1997) "Geographical Information Systems and Computer Cartography" Addison Wesley Longman, Essex, England.
46. Lee, R.C., and Moore, J.M. (1967) "CORELAP: Computerized Relationship Layout Planning." J. Industrial Engrg., 8(3), 195-200.
47. Li, H., and Love, P.E.D. (1998), "Site-level Facilities Layout Using Genetic Algorithms", J. of Computing in Civil Engrg., ASCE, 12(4), 227-231.
48. Luger, G. F., and Stubblefield, W.A. (1998), "Artificial Intelligence and the Design of Expert Systems", Benjamin/Cummings Pub.
49. Lundberg, E.J., and Beliveau, Y.J. (1989) "Automated Lay-Down Yard Control System- ALYC" J. of Construction Engrg. and Management, ASCE, 115(4), 535-544.

50. Mahoney, J.J., and Tatum, B.C., (1994) "Construction Site Applications of CAD" J. of Construction Engrg. and Management, ASCE, 120(3), 116-124.
51. Moore, J. (1971), "Computer Program Evaluates Plant Layout Alternatives", Industrial Engineering, 3(80), 19-25.
52. Moore, J. (1980), "Computer Methods in Facility Layout", Industrial Engineering, 5(89), 82-93.
53. Mawdesley, M.J., Al-jibouri, S. H., and Yang, H. (2002). "Genetic Algorithm for Construction Site Layout in Project Planning" J. of Construction Engrg. and Management, ASCE, 128(5), 418-426.
54. McKinney, K., and Fischer, M. (1997) "4D analysis of Temporary Support" Proc. of the 4th Congress in conj. with A/E/C, Computing in Civil Engrg., ASCE, Philadelphia, Pennsylvania, 248-256.
55. McKinney, K., Kim, J., Fischer, M., and Howard, C. (1996) "Interactive 4D CAD" Proc. of the 3rd Congress in conj. with A/E/C, Computing in Civil Engrg., ASCE, Anaheim, CA, 383-389.
56. Philip, M., Mahadevan, N., and Varghese, K. (1997) "Optimization of Construction Site Layout – A Genetic Algorithm Approach" Proc. of 4th ASCE Congress on Computing in Civil Engrg., Philadelphia, PA, 710-718.
57. Rad, P.F. (1980) "Analysis of Working Space Congestion from Scheduling Data", Transactions of American Association of Cost Engineering, AACE, Washington D.C., F.4.1-F.4.5.

58. Rad, P.F. (1982) "A Graphic Approach to Construction Job-Site Planning" *Cost Engineering*, AACE, 24(4), 211-217.
59. Rad, P.F. and James, B.M. (1983) "The Layout of Temporary Construction Facilities" *Cost Engineering*, AACE, 25(2), 19-27.
60. Riley, D.R. (1998) "4D Space Planning Specification Development for Construction Work Spaces" *Proc. of International Computing Congress held in Conj. with ASCE Annual Convention and Exhibition*, Boston, Massachusetts, 354-363.
61. Riley, D.R. (1995) "Space Planning Applications in Multi -Story Buildings", *Construction Congress, Proc. of '95 Conference*, ASCE, San Diego, CA, 339-346.
62. Riley, D.R., and Sanvido, V.E. (1997), "Space Planning Method for Multistory Building Construction", *J. of Construction Engrg. and Management*, ASCE, 123(2), 171-180.
63. Riley, D.R., and Tommelein, I.D. (1996) "Space Planning Tools for Multi-story Construction" *Proc. 3th Congress held in Conj. with A/E/C Systems*, ASCE, Anaheim, CA, 718-725.
64. Rivard, H., and Fenves, S.J. (2000), "A Representation for Conceptual Design of Buildings", *J. of Computing in Civil Engrg.*, ASCE, 14(3), 151-159.
65. Sadeghpour, F., Moselhi, O., and Alkass, S. (2002). "Dynamic Planning for Site Layout", *Proc. of 30th Annual Conference of Canadian Society of Civil Engrg.*, CSCE, Montreal, Canada.

66. Sadeghpour, F., Moselhi, O., and Alkass, S. (2003). "Open Architecture for Site Layout Modeling", *Proc. of International Symposium of Automation and Robotics in Construction*, ISARC, Eindhoven, Netherlands.
67. Sadeghpour, F., Moselhi, O., and Alkass, S. (2004)a. "Geometric Reasoning for Site Space Analysis", *Proc. of World Building Congress*, CIB, Toronto, Canada.
68. Sadeghpour, F., Moselhi, O., and Alkass, S. (2004)b. "A CAD-Based Model for Site Planning", *Journal of Automation in Construction*, Elsevier, (in press).
69. Sadeghpour, F., Moselhi, O., and Alkass, S. (2004)c. "Graphical Constraint Representation for Site Layout", *Proceedings of the 4th International Conf. on Construction Applications of Virtual Reality*, Lisbon, Portugal (in press).
70. Scribin, M., and Vergin, R.C. (1975) "Comparison of Computer Algorithm and Visual Based Methods for Plant Layout", *Management Science*, 22(2), 172-181.
71. Simon (1996) "The science of the artificial" 3rd Ed., MIT Press, Cambridge, Mass.
72. Sirinaovakul, B., and Thajchayapong, P. (1996) "An Analysis of Computer-Aided Facility Layout Techniques" *J. of Computer-Aided Manufacturing*, 9(4), 260-264.
73. Smith, I.F.C., Kurmann, D., and Schmitt, G. (1995) "Case Combination and Adaptation of Building Spaces" *Proc. 2nd Congress held in Conj. with A/E/C Systems '95*, Atlanta, GA, 155-162.
74. Staub, S., and Fischer, M. (1998) "Constructability Reasoning Based on a 4D Facility Model", *SEI-ASCE 13th Conference of Analysis and Computation, Structural Engineering World Wide*, Elsevier, Paper Ref. T191-1.

75. Sutphin, J. (1999) "AutoCAD 2000 VBA Programmer's Reference", Wrox Press, Brimingham.
76. Tam, C.M., and Tong, T.K.L. (2003). "GA-ANN model for optimizing the locations of tower crane and supply points for high-rise public housing construction" *Construction Management and Economics*, Spon Press, 21(3), 257-266.
77. Tam, C.M., Tong, T.K.L., and Leung, A.W.T., Chiu, G.W.C. (2002). "Site Layout Planning Using Nonstructural Fuzzy Decision Support System" *J. of Construction Engrg. and Management*, ASCE, 128(3), 220-231.
78. Thabet, W.Y., and Beliveau, Y. J. (1993) "A Model to Quantify Work Space Availability for Space-Constrained Scheduling within a CAD Environment", *Proc. 5th Conference in Computing in Civil and Building Engrg.*, ASCE, New York, NY, 110-116.
79. Thabet, W.Y., and Beliveau, Y. J. (1994)a "Alternative Decision for Space-Based Scheduling in Multi-Story Projects", *Proc. 1st Congress held in conj. with A/E/C Systems, Computing in Civil Engrg.*, ASCE, Washington D.C., 1164-1171.
80. Thabet, W.Y., and Beliveau, Y. J. (1994)b "Modeling Work Space to Schedule Repetitive Floors in Multistory Buildings", *J. of Construction Engineering and Management*, ASCE, 120(1), 96-116.
81. Thabet, W.Y., and Beliveau, Y. J. (1997) "ScaRC: Space-Constrained Resource Constrained Scheduling System", *J. of Computing in Civil Engineering*, ASCE, 11(1), 48-59.
82. Thabet, W.Y., Morad, A.A., and Beliveau, Y. J. (1992) "Work Space Constraints Modeling for Process Scheduling Using Artificial Intelligence and 3D Computer

- Modeling Technologies”, Proc. 8th Conference held in Conj. with A/E/C Systems ‘92, Computing in Civil Engrg. and Geographic Information System Symposium, Dallas, Texas, 727-737.
83. Tommelein, I.D. (1991) “Site Layout: Where Should It Go?” Proc. of Construction Congress ‘91, ASCE, Cambridge, MA. 632-637.
 84. Tommelein, I.D. (1992) “Construction Site Layouts Using Blackboard Reasoning with Layered Knowledge” Expert Systems for Civil Engineers: Knowledge Representation, ASCE, 214-259.
 85. Tommelein, I.D. (1994)a “Materials Handling and Site Layout Control” Proc. 11th International Symposium in Construction (Automation and Robotics in Construction XI), ISARC, Brighton, UK, 297-304.
 86. Tommelein, I.D. (1994)b “MoveCapPlan: An Integrated System for Planning and Controlling Construction Material Laydown and Handling” Proc. 1st Congress held in Conj. with A/E/C Systems, ASCE, Washington D.C., 1172-1179.
 87. Tommelein, I.D. (1995) “New Tools for Site Material Handling and Layout Control” Construction Congress, Proc. of ‘95 Conference, ASCE, San Diego, CA, 479-487.
 88. Tommelein, I.D., Dzeng, R.J., and Zouein, P.P. (1993) “Exchanging Layout and Schedule Data in a Real-Time Distributed Environment” Proc. 5th Conference Comp. in Civil Engrg., Anaheim, CA, 947-954.
 89. Tommelein, I.D., Gastillo, J.G., and Zouein, P.P. (1992)a “Space-Time Characterization for Resource Management on Construction Sites” Proc. 8th Conference Comp. in Civil Engrg., Dallas, Texas, 623-630.

90. Tommelein, I.D., Levitt, R.E., and Hayes-Roth, B. (1992)b "How can Artificial Intelligence Help?" J. of Construction Engrg. and Management, ASCE, 118(3), 594- 611.
91. Tommelein, I.D., Levitt, R.E., and Hayes-Roth, B. (1992)c "SightPlan Model for Site Layout" J. of Construction Engrg. and Management, ASCE, 118(4), 749-766.
92. Tommelein, I.D., Levitt, R.E., Hayes-Roth, B., and Confery, T. (1991) "SightPlan Experiments: Alternate Strategies for Site Layout Design" J. of Computing in Civil Engrg., ASCE, 5(1), 42-63.
93. Tommelein, I.D., and Zouein, P.P. (1992) "Activity Level Space Scheduling", Proc. of 9th International Symposium on Automation and Robotics in Construction, ISARC, Tokyo, Japan, 411-420.
94. Tommelein, I.D., and Zouein, P.P. (1993)a "Interactive Dynamic Layout Planning" J. of Construction Engrg. and Management, ASCE, 119(2), 266-287.
95. Tommelein, I.D., and Zouein, P.P. (1993)b "Space Scheduling for Construction Progress Planning and Control", Proc. of 10th International Symposium on Automation and Robotics in Construction, Houston, TX, 415-422.
96. Toole, T., M. (2002) "Construction Site Safety Roles", Journal of Construction Engrg. and Management, 128(3), 203-210.
97. Wasserman, P. D. (1989) "Neural Computing: Theory and Practice", Van Nostrand Reinhold, New York, NY.
98. Yeh, I.C. (1995) "Construction-Site Layout Using Annealed Neural Network" J. of Computing in Civil Engrg., ASCE, 9(3), 201-208.

99. Zamanian, M.K. (1992), "Modeling and Communicating Spatial and Functional Information about Constructed Facilities" Ph.D. Dissertation, Engineering Design Research Center, Carnegie Mellon University, Pittsburgh, PA.
100. Zouein, P.P. (1996) "MoveSchedule: A Planning Tool for Scheduling Space use on Construction Sites" Ph.D. Dissertation, Univ. of Michigan, Anna Arbor, MI.
101. Zouein, P.P., Harmanani, H., and Hajar, A. (2002) "Genetic Algorithm for Solving Site Layout Problem with Unequal-Size and Constrained Facilities" J. of Computing in Civil Engrg., ASCE, 16(2), 143-151.
102. Zouein, P.P., and Tommelein, I.D. (1993) "Space Schedule Construction Sites" Proc. 5th Conference Comp. in Civil and Building Engrg., Anaheim, CA, 1770-1777.
103. Zouein, P.P., and Tommelein, I.D. (1994)a "Time-Space Tradeoff Strategies for Space-Schedule Construction " Proc. 1st Conference Comp. in Civil Engrg. in Conj. with A/E/C Systems, Washington D.C., 1180-1187.
104. Zouein, P.P., and Tommelein, I.D. (1994)b "Automating Dynamic Layout Construction" Proc. 11th International Symposium in Construction (Automation and Robotics in Construction XI), ISARC, Brighton, UK, 409-416.
105. Zouein, P.P., and Tommelein, I.D., (1999) "Dynamic Layout Planning Using a Hybrid Incremental Solution Method" J. of Construction Engrg. and Management, ASCE, 125(6), 400-408.
106. Zouein, P.P., and Tommelein, I.D., (2001) "Improvement Algorithm for Limited Space Scheduling" J. of Construction Engrg. and Management, ASCE, 127(2), 116-124.

APPENDIX I

I.I SITE OBJECTS

1. Site boundary
2. Site edges
3. Service line- Electricity
4. Service line-Water supply
5. Service line- Gas line
6. Service line- Telephone line
7. Service line- Sewage
8. Access road- Temporary
9. Access road- Permanent
10. Railroad
11. Site entrance- car
12. Site entrance- equipment
13. Marked area- unavailable
14. Marked area- hazardous
15. Marked area- underground activity
16. Marked area- specific terrain type
17. Permanent facility- existing structures and utilities
18. Permanent facility- building under construction
19. Natural features- tree
20. Natural features- water pound

I.II TEMPORARY FACILITIES

Most commonly used temporary facilities (Rad 1982).

- | | |
|--|--|
| 1. Job office | 24. Laydown areas- lumber (long-term) |
| 2. Craft change house- civil | |
| 3. Craft change house- mechanical | 25. Laydown areas- lumber (short-term) |
| 4. Craft change house- piping | 26. Test shop- concrete |
| 5. Craft change house- electrical | 27. Test shop- other materials |
| 6. Time office | 28. Equipment maintenance and storage shop |
| 7. Brass alleys | 29. Test shop- welding |
| 8. Sanitary facilities | 30. Parking lot- crafts |
| 9. Warehouse- electrical supplies | 31. Parking lot- office |
| 10. Warehouse- mechanical supplies | 32. Parking lot- owner |
| 11. Warehouse- instrumentation supplies | 33. Access roads- temporary |
| 12. Warehouse- office | 34. Access roads- permanent |
| 13. Fabrication shop- electrical | 35. Batch plants |
| 14. Fabrication shop- mechanical | 36. Railroad |
| 15. Fabrication shop- carpentry | 37. Paint shops |
| 16. Material staging area- general | 38. Sandblasting |
| 17. Material staging area- electrical materials | 39. Living facilities- camp |
| 18. Material staging area- mechanical materials | 40. Living facilities- trailer |
| 19. Material staging area- instrumentation materials | 41. Medical and first aid facility |
| 20. Laydown areas- structural (long-term) | 42. Rebar shop |
| 21. Laydown areas- structural (short-term) | 43. Main gate guard house |
| 22. Laydown areas- concrete pipe (long-term) | 44. Payroll office |
| 23. Laydown areas- concrete pipe (short-term) | 45. Water treatment plant |
| | 46. Pump house |
| | 47. Explosive storage shed |
| | 48. Ice plant |

I.III LOCATING CONSTRAINTS

A set of constraints compiled from personal communications with site personnel (engineer, manager, superintendents), and safety code of for construction in Quebec (ASP Construction 2001). The weight column expresses the relevant preference of a constraint over others in a scale of 1 to 5, with 5 showing the highest desirability and 1 representing the least.

Temporary Facility	Constraint	Weight	Notes
Offices	Access to service lines.	5	
	Close to other offices (contractor, owner).	3	If there are several offices on site, it is preferred to keep them close together due to travel frequency among them.
	Within 100m of office parking.	4	
Sanitary facilities	Trailer: Close to craft change house.	4	
	Portable: Near work areas.	5	Within 5 minutes walking distance.
	Accessible by suction truck.	5	Cleaning is required 2-3 times a week.
Craft change	Close craft to parking.	3	On projects where underground activities are involved, craft change house is equipped with a mounting tool that lifts the craft clothes to a high level to dry overnight.
	Close to service lines.	4	
	Close to sanitary trailer.	4	
Warehouses (Electrical, mechanical, instrumentation)	Close to other warehouses.	3	Often warehouse of different purpose are grouped under the same structure.
	Close to access road.	3	
	Close to electricity line.	4	Depending on what they store, warehouses can be heated or unheated. For example warehouse for precise tools and instruments requires heating.
	Close to respective workshop.	4	
Test shops/onsite laboratories (Concrete, soil, welding)	Close to offices.	2	There is a lot of paper work and communication between test shops and site office.
	Close to service lines.	3	
	Close to garage.	4	

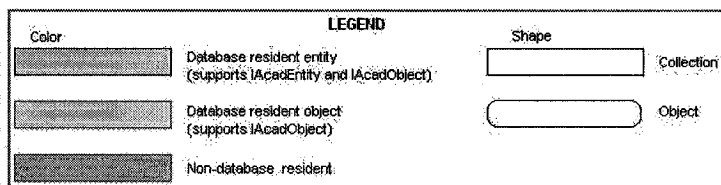
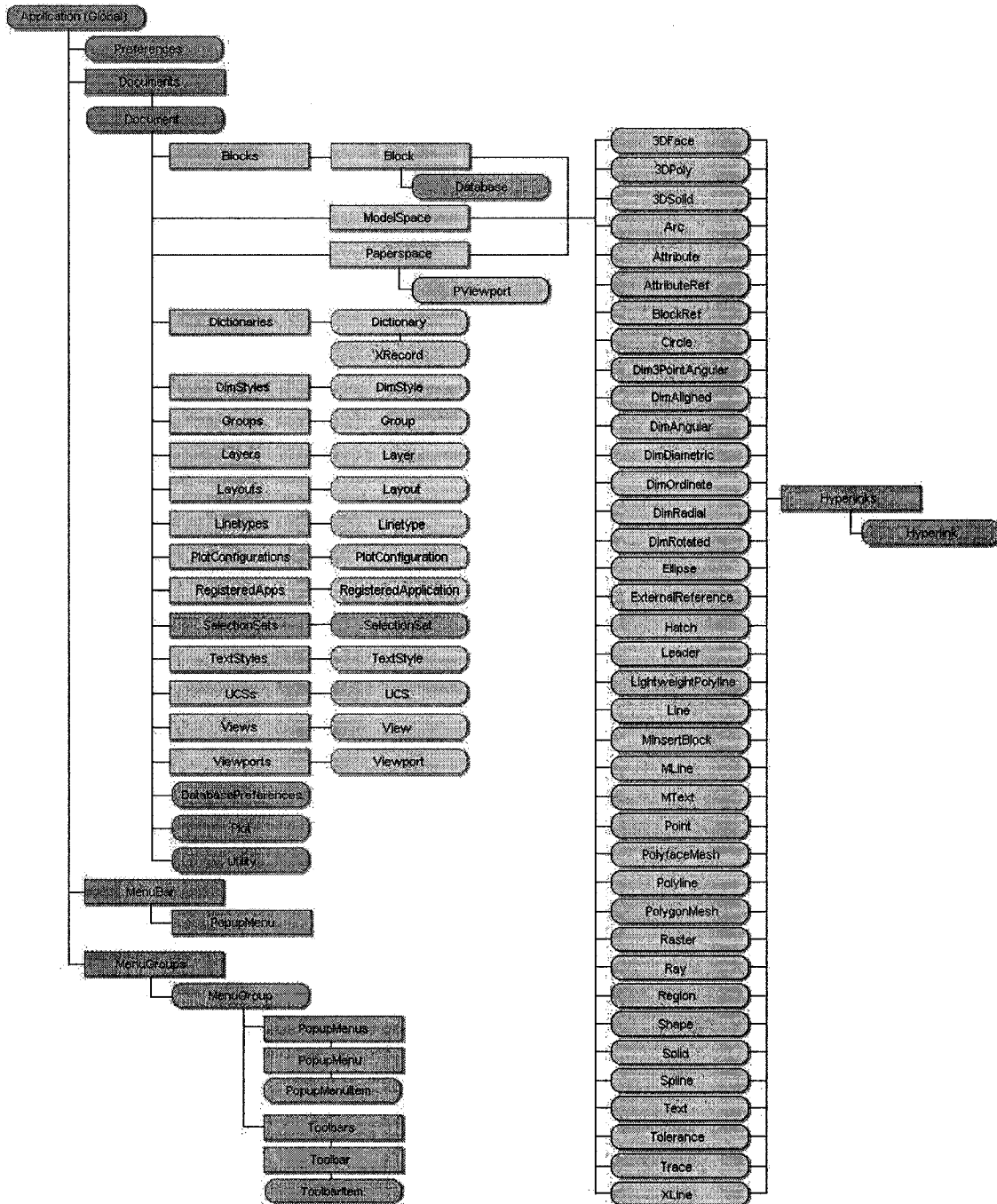
Temporary Facility	Constraint	Weight	Notes
Office warehouse (storage)	Close to other warehouses.	2	
	Close to access road.	3	
	Close to electricity line.	4	
	Visible from office.	5	For security reasons and to trace information.
Staging areas (General, electrical, mechanical)	In an area with solid ground.	4	Material should be organized so that an access path inside the area is maintained.
	Close to relative workshop.	2	
Workshops (Electrical and mechanical)	Close to other workshops.	3	To facilitate sharing of equipment and tools.
	Close to electricity line.	4	
	Close to pressurized air line.	4	To compress the air.
	Far from offices.	5	To keep the noise away.
	Close to respective warehouse.	4	
Carpentry workshop	Adjacent to formwork storage area.	5	Easy access by lifts should be considered.
	Close to electricity line.	5	
Material laydown areas (Structural, concrete pipe, lumber)	Prefabricated: close to construction.	2	Since the materials are heavy, it is desirable to move them as less as possible.
	Fabricated on site: close to respective workshop.	4	
	Within reach of crane.	5	
Garage	Not under electricity line.	5	For safety reasons; to reduce the risk for the equipment with high parts.
	Close to service lines.	3	
	Close to road.	3	
	Within 100m of entrance.	4	
	Close to mechanical and electrical shops.	4	

Temporary Facility	Constraint	Weight	Notes
Parking lots (craft, contractor, owner)	Close to the respective offices.	3	
Batch plants	Close to water and electricity lines.	3	
	Close to construction.	4	
	Next to water treatment plant.	5	See water treatment plant below.
Paint and sandblast shops	Close to steel fabrication shop.	3	
	Far from offices.	4	Due to dust and noise.
Living facilities	Close to entertainment and catering facilities.	4	
	Far from construction.	5	To avoid construction noise.
Medical and first aid facility	Close to offices.	4	In case of accidents, office is informed first.
	Close to service lines.	3	
Rebar shop	Adjacent to rebar staging area.	4	
	Within 40m of construction.	5	Not desired to move rebar in long distances.
Explosive storage shed	Far from everything on site.	5	Usually a secondary access road is built for explosives.
	Not within 200m of anything.	5	Explosive storage should be surrounded by dikes.
	Far from road.	5	
Water treatment plant	Close to swage ditch.	4	For environmental reasons, waste water used to clean the concrete mixer has to be treated before being released to sewage system.
	Next to concrete batch plant.	5	
Pump house	Close to waterbed.	5	Pump house is used when a large amount of water has to be removed. Otherwise portable pumps are sufficient for smaller jobs.
	Close to electricity line.	4	
	Close to swage ditch.	4	

Temporary Facility	Constraint	Weight	Notes
Guardhouse	Next to main entrance.	5	
	Visible from offices.	4	
	View to storage.	5	For security reasons.
	View to access road.	4	For security reasons.

APPENDIX II

II.I AUTOCAD GRAPHICAL OBJECTS PROGRAMMING INTERFACE (AUTODESK 2001)



II.II AUTOCAD OBJECT DESCRIPTION

POLYLINE OBJECT

VBA object name:	AcadPolyline
Create using:	ModelSpace.AddPolyline PaperSpace.AddPolyline Block.AddPolyline
Access via:	ModelSpace.Item PaperSpace.Item Block.Item Group.Item SelectionSet.Item
Automation Interfaces Supported:	IacadPolyline IAcadEntity IAcadObject

Polyline Properties	Polyline Methods	Polyline Events
Application	AppendVertex	Mirror
Area	ArrayPolar	Mirror3D
Closed	ArrayRectangular	Move
Color	Copy	Offset
Coordinate	Delete	Rotate
Coordinates	Explode	Rotate3D
Document	GetBoundingBox	ScaleEntity
Elevation	GetBulge	SetBulge
Hyperlinks	GetExtensionDictionary	SetWidth
Handle	GetWidth	SetXData
HasExtensionDictionary	GetXData	TransformBy
Layer	Highlight	Update
Linetype	IntersectWith	

REGION OBJECT

VBA object name:	AcadRegion
Create using:	ModelSpace.AddRegion PaperSpace.AddRegion Block.AddRegion
Access via:	ModelSpace.Item PaperSpace.Item Block.Item SelectionSet.Item Group.Item
Automation Interfaces Supported:	IAcadRegion IAcadEntity

Region Methods	Region Properties		Region Events
ArrayPolar	Rotate3D	Application	Modified
ArrayRectangular	ScaleEntity	Area	
Boolean	SetXData	Centroid	
Copy	TransformBy	Color	
Delete	Update	Document	
Explode		Handle	
GetBoundingBox		HasExtensionDictionary	
GetExtensionDictionary		Hyperlinks	
GetXData		Layer	
Highlight		Linetype	
IntersectWith		LinetypeScale	
Mirror		Lineweight	
Mirror3D		MomentOfInertia	
Move		Normal	
Rotate		ObjectID	
		ObjectName	
		OwnerID	
		Perimeter	
		PlotStyleName	
		PrincipalDirections	
		PrincipalMoments	
		ProductOfInertia	
		RadiiOfGyration	
		Visible	

BLOCK OBJECT

VBA object name:	AcadBlock
Create using:	Blocks.Add
Access via:	Blocks.Item Layout.Block
Automation Interfaces Supported:	IAcadBlock IAcadObject IacadObjectEvents

Block Methods	Block Methods			Block Events
Add3Dface	AddExtrudedSolidAlongPath	AddText	Application	Modified
Add3Dmesh	AddExtrudedSolid	AddTolerance	Count	
Add3Dpoly	AddHatch	AddTorus	Document	
AddArc	AddLeader	AddTrace	Handle	
AddAttribute	AddLightweightPolyline	AddWedge	HasExtensionDictionary	
AddBox	AddLine	AddXLine	IsLayout	
AddCircle	AddMInsertBlock	AttachExternalReference	IsXRef	
AddCone	AddMLine	Bind	LayoutName	
AddCustomObject	AddMText	Delete	ObjectID	
AddCylinder	AddPoint	Detach	Origin	
AddDim3PointAngular	AddPolyfaceMesh	Delete	OwnerID	
AddDimAligned	AddPolyline	Detach	XRefDatabase	
AddDimAngular	AddRaster	GetXData		
AddDimDiametric	AddRay	InsertBlock		
AddDimOrdinate	AddRegion	Item		
AddDimRadial	AddRevolvedSolid	Reload		
AddDimRotated	AddShape	SetXData		
AddEllipse	AddSolid	Unload		
AddEllipticalCone	AddSphere			
AddEllipticalCylinder	AddSpline			

BLOCKREF OBJECT

VBA object name:	AcadBlockReference
Create using:	ModelSpace.InsertBlock PaperSpace.InsertBlock Block.InsertBlock
Access via:	ModelSpace.Item PaperSpace.Item Block.Item SelectionSet.Item Group.Item

BlockRef Methods	BlockRef Properties		BlockRef Events
ArrayPolar	Rotate	Application	Modified
ArrayRectangular	Rotate3D	Color	OwnerID
Copy	ScaleEntity	Document	PlotStyleName
Delete	SetXData	Hyperlinks	Rotation
Explode	TransformBy	Handle	Visible
GetAttributes	Update	HasAttributes	XscaleFactor
GetBoundingBox		HasExtensionDictionary	YscaleFactor
GetConstantAttributes		InsertionPoint	ZScaleFactor
GetExtensionDictionary		Layer	
GetXData		Linetype	
Highlight		LinetypeScale	
IntersectWith		Lineweight	
Mirror		Name	
Mirror3D		Normal	
Move		ObjectID	

APPENDIX III

SITE IMAGERY OF THE LG-2 PROJECT



Figure III.1 View of LG-2 project site



Figure III.2 Spillway



Figure III.3 South-east view of the site

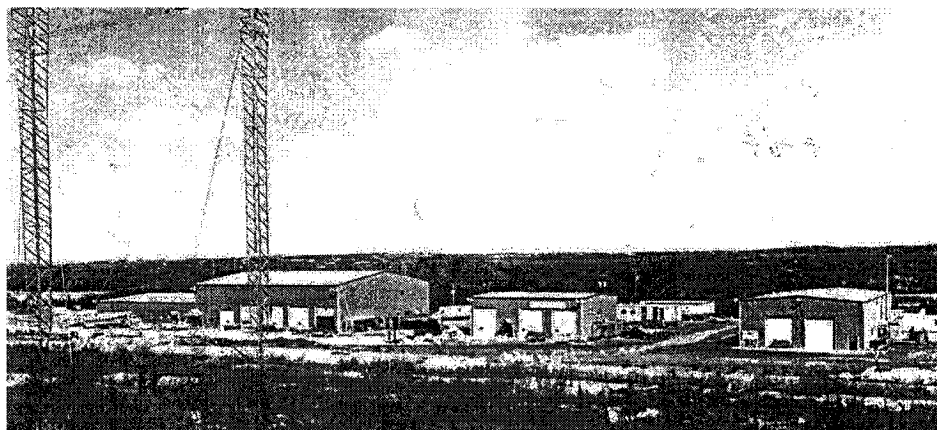


Figure III.4 South view of the site

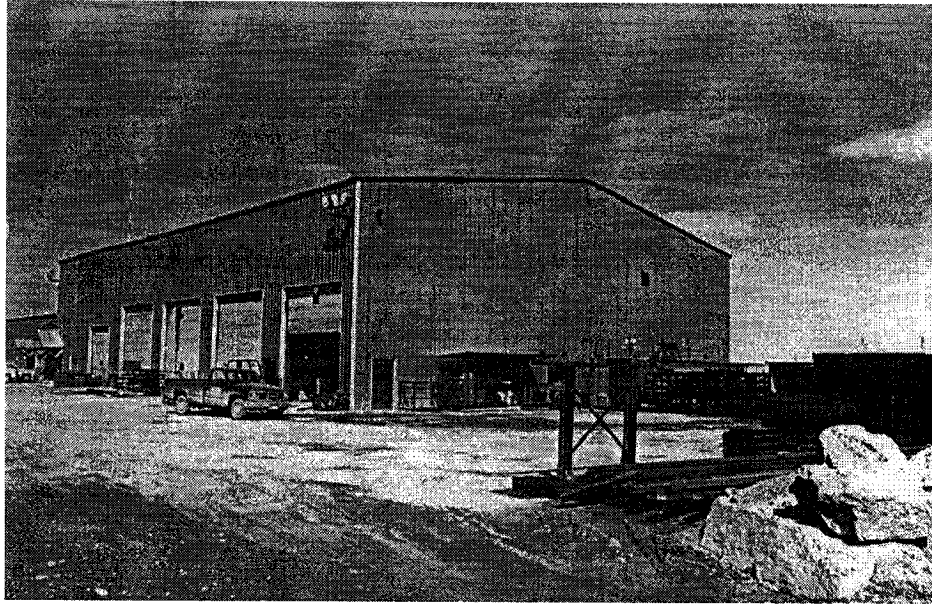


Figure III.5 View of garage

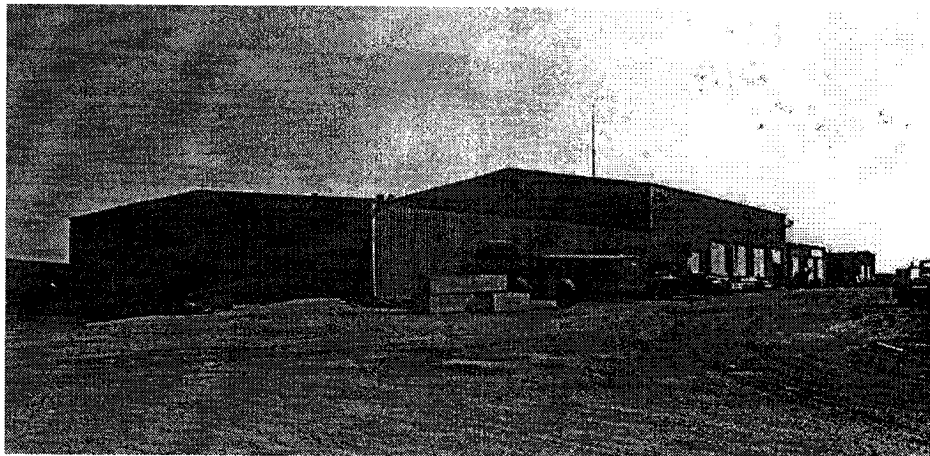


Figure III.6 View of storage, garage, and equipment parking