A Hierarchical Evolutionary Algorithmic Design (HEAD) System for Generating and Evolving Building Design Models

by

Fakhri A Bukhari
MArch., MAUD

A thesis submitted in partial fulfilment of the requirements for the degree of

Doctor of Philosophy

Built Environment and Engineering, School of Design

Queensland University of Technology

2011

Abstract

This thesis develops a detailed conceptual design method and a system software architecture defined with a parametric and generative evolutionary design system to support an integrated interdisciplinary building design approach. The research recognises the need to shift design efforts toward the earliest phases of the design process to support crucial design decisions that have a substantial cost implication on the overall project budget.

The overall motivation of the research is to improve the quality of designs produced at the author's employer, the General Directorate of Major Works (GDMW) of the Saudi Arabian Armed Forces. GDMW produces many buildings that have standard requirements, across a wide range of environmental and social circumstances. A rapid means of customising designs for local circumstances would have significant benefits. The research considers the use of evolutionary genetic algorithms in the design process and the ability to generate and assess a wider range of potential design solutions than a human could manage. This wider ranging assessment, during the early stages of the design process, means that the generated solutions will be more appropriate for the defined design problem.

The research work proposes a design method and system that promotes a collaborative relationship between human creativity and the computer capability. The tectonic design approach is adopted as a process oriented design that values the process of design as much as the product.

The aim is to connect the evolutionary systems to performance assessment applications, which are used as prioritised fitness functions. This will produce design solutions that respond to their environmental and function requirements. This integrated, interdisciplinary approach to design will produce solutions through a design process that considers and balances the requirements of all aspects of the design.

Since this thesis covers a wide area of research material, 'methodological pluralism' approach was used, incorporating both prescriptive and descriptive research methods. Multiple models of research were combined and the overall research was undertaken following three main stages, conceptualisation, developmental and evaluation. The first two stages lay the foundations for the specification of the proposed system where key aspects of the system that have not previously been proven in the literature, were implemented to test the feasibility of the system. As a result of combining the existing knowledge in the area with the newly-verified key aspects of the proposed system, this research can form the base for a future software development project. The evaluation stage, which includes building the prototype system to test and evaluate the system performance based on the criteria defined in the earlier stage, is not within the scope this thesis.

The research results in a conceptual design method and a proposed system software architecture. The proposed system is called the 'Hierarchical Evolutionary Algorithmic Design (HEAD) System'. The HEAD system has shown to be feasible through the initial illustrative paper-based simulation. The HEAD system consists of the two main components - 'Design Schema' and the 'Synthesis Algorithms'. The HEAD system reflects the major research contribution in the way it is conceptualised, while secondary contributions are achieved within the system components.

The design schema provides constraints on the generation of designs, thus enabling the designer to create a wide range of potential designs that can then be analysed for desirable characteristics. The design schema supports the digital representation of the human creativity of designers into a dynamic design framework that can be encoded and then executed through the use of evolutionary genetic algorithms.

The design schema incorporates 2D and 3D geometry and graph theory for space layout planning and building formation using the Lowest Common Design Denominator (LCDD) of a parameterised 2D module and a 3D

structural module. This provides a bridge between the standard adjacency requirements and the evolutionary system. The use of graphs as an input to the evolutionary algorithm supports the introduction of constraints in a way that is not supported by standard evolutionary techniques. The process of design synthesis is guided as a higher level description of the building that supports geometrical constraints.

The Synthesis Algorithms component analyses designs at four levels, 'Room', 'Layout', 'Building' and 'Optimisation'. At each level multiple fitness functions are embedded into the genetic algorithm to target the specific requirements of the relevant decomposed part of the design problem. Decomposing the design problem to allow for the design requirements of each level to be dealt with separately and then reassembling them in a bottom up approach reduces the generation of non-viable solutions through constraining the options available at the next higher level. The iterative approach, in exploring the range of design solutions through modification of the design schema as the understanding of the design problem improves, assists in identifying conflicts in the design requirements. Additionally, the hierarchical set-up allows the embedding of multiple fitness functions into the genetic algorithm, each relevant to a specific level. This supports an integrated multi-level, multi-disciplinary approach.

The HEAD system promotes a collaborative relationship between human creativity and the computer capability. The design schema component, as the input to the procedural algorithms, enables the encoding of certain aspects of the designer's subjective creativity. By focusing on finding solutions for the relevant sub-problems at the appropriate levels of detail, the hierarchical nature of the system assist in the design decision-making process.

Keywords

Architecture, Conceptual Design, Computer Aided Architectural Design (CAAD), Design Schema, Design System, Design Method, Design Process, Evolutionary Architecture System, Genetic Algorithms, Generative Design, Modularity

Dedication

This thesis is dedicated to my role model and mentor,

my late father, General / AbdulGhani Bukhari.

May God bless his soul.

Acknowledgements

I am very grateful for all the knowledge I have received throughout these years of PhD studies. It is a pleasure to thank all those who directly made this thesis possible.

I thank Professor John Frazer, my supervisor for enlightening me through his wide knowledge and his deep intuition. Working closely with John has been a privilege and I am grateful for the opportunity I was given to learn from his vast experience. I also thank Professor Robin Drogemuller, my associate supervisor, for his continual support and help, especially in coming to grips with the technicalities involved in my area of research.

During this research work, I have collaborated with many colleagues for whom I have great regard, and I extend my warmest thanks to all those who have helped me with my work, in particular, the VIBES research group.

To the General Directorate of Military Works (GDMW) of the Saudi Arabian Armed Forces, I extend my deep gratitude for not only granting me the opportunity to pursue my education further, but for supporting me throughout. Although the data provided by the GDMW is subject to military confidentiality (especially for publishing), it was crucial to my research and I am very grateful to my colleagues for allowing me to use it.

I am indebted to my parents for teaching me to value the pursuit of knowledge since my early childhood. I hope that I can be a role model for my daughter and son, the same way that my parents were to me.

This thesis has been edited by Tony Roberts.

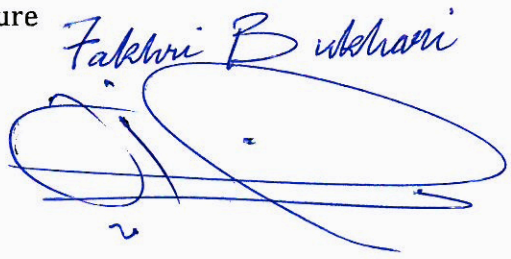
Contents

List of Figures

List of Tables

Authorship

The work contained in this thesis has not been previously submitted to meet requirements for an award at this or any other higher education institution. To the best of my knowledge and belief, the thesis contains no material previously published or written by another person except where due reference is made.

Signature 

Date 15/11/2011

1 Introduction

1.1 Overview of problem

The overall research project develops a design method and system for generating architectural building designs. This thesis sets the foundation for the overall research project resulting in the blue print of conceptual system architecture where a prototype system will be built accordingly. The research targets the initial conceptual stage of the design process as the major influential design decisions are made. This research focuses on four areas of relevance and contribution, Creativity in Design, Algorithmic Architectural Design, Parametric and Modular Design and Integrated Interdisciplinary Design.

This research seeks to express the central role, played by the designer's human cognitive thinking process and creativity, within the Computer Aided Architectural Design (CAAD) process through the application of algorithmic design methods, such as the generative design and evolutionary design approaches. Algorithmic design approaches have the capacity to incorporate parameterised building components into the propagation cycles governed by a set of rules introduced by the designer to evolve building design solutions. Additionally, algorithmic design approaches have the means to integrate an interdisciplinary design process through the constant evaluation and assessment of the generated design solutions using fitness functions, assigned by the designer that encode factors from multiple disciplines.

On the other hand, although the modular approach in design, which came to the fore during the industrial revolution of the 19th century, has been applied to the mass production of some building complexes, it challenges two essential codes of the architecture and design practice. Firstly, it is found to contradict, to some extent, the fundamental architectural principle of the uniqueness of each project that usually leads to a less performant building. Secondly, it conflicts with the artistic freedom that designers

strenuously defended and fought to maintain during the design rationality movement. The parametric design approach include the new notion of modularity in design that embraces and respects the designer's creative and artistic visions in addition to the uniqueness of each project in relation to its environment.

The complexity inherent in building design and construction requires an integrated interdisciplinary design approach to the architectural design process. For a building's design solution to respond to the functional, social, cultural and economic needs of its users and environment, all the different disciplines involved in the design of the building, (which includes the various systems of structural, mechanical, electrical and architectural designs), must be integrated into a multidisciplinary design process that considers all aspects of the building design solution. Although the integrated interdisciplinary design is seen as a novel and promising approach to design, it has not been achieved successfully. In current practice, designers from the various disciplines usually begin their participation in the design process after the initial architectural conceptual design is produced.

Evolutionary architectural design systems produce design solutions in response to the targeted performance requirements formulated as fitness functions fed into its genetic algorithmic engine. The evolutionary system can host a number of fitness functions that can reflect the interdisciplinary requirements of the architectural building design. Linking of the building performance simulations and animations to the evolutionary design process generates more responsive design solutions. The inevitable conflicts between different design constraints and requirements can be resolved through the prioritisation of the design constraints and requirements.

This research, therefore, develops the HEAD system's architecture based on parametric and generative evolutionary design methods with the aim of

supporting integrated building design and promoting a 'collaborative relationship between human creativity and computational capability'.

1.1.1 Design thinking and creativity

When developing computer design systems, one is obliged to understand how designers think and how the design evolves. Although many respected researchers have provided a general understanding of their characteristics, both human creativity and the human thinking processes in design have been recognized as mysterious and indeterminate. Nevertheless, this research recognises the designer's preconceptions which Broadbent (1988) introduced, as the source of the designer's creative inputs during the synthesis of design.

Most definitions of creativity 'are often misleading: they say too much and too little' (Taylor, 2009, p. 2). In a similar vein, it is apparent that the 'battles over the correct definition of design are fruitless' and 'the field will be vital as long as definitions come and go in the work of inquiry' (Buchanan, 2004, p. 15). The design methods movement of the early 1960's made an attempt to make the realm of design more 'scientific', based on science, technology and rationalism (Cross, 1993, 2007).

By the early 1970s Broadbent (1988) included a designer's 'pre-conceptions' to the synthesis stage of the design process, thus modifying the design process of the 1960s. The 1980s saw the emergence of an interest in symbolic forms and functionalism in architecture. The scientific, rationalistic approach to design was almost completely rejected. Design, as argued by Cross et al. (1981) 'simply is not like science' and should instead 'be viewed as a technological activity' (Cross et al., 1981, p. 200). According to Cross (1982, p. 222):

> … technology involves a synthesis of knowledge and skills from both the sciences and the humanities, in the pursuit of practical tasks; it is not simply 'applied science', but 'the application of scientific and organized knowledge to practical tasks.

The designer's preconceptions, which Broadbent (1988) argued were a vital part of the synthesis of design, brings the designer's role in the design process to the fore. Three types of preconceptions, which will be discussed in greater detail in Section (3.1.3.3) and Section (4.5), can be identified.

- Guiding Principles
- Primary Generators
- Design Schema

A differentiation is made within this thesis between the design schema, which targets a specific project and the designer's schema (personal style), which is the accumulation of the common design features of a designer which result from a long practicing career. The design schema will be a specialisation of the designer's schema, refined to suit the problem at hand. The designer's and the design schema are discussed in Sections (4.5.3) and (4.6).

1.1.2 Algorithmic architectural design

Over the years, the design process has changed and developed from 'design through production' involving 'craft' or 'blacksmith designs', to 'design through drafting' — and again, through the struggle to define creativity in design and the embracing of computers in the design process, to their rejection and ultimately, the acceptance of the computer once again. Today, the role of the designer in the design process is closely related to the role of the computer in the design framework (Janssen, Frazer & Tang, 2002). As Terzidis (2006) points out, 'it is apparent that design is strongly dependent on the tools utilized and, reversely, tools have a profound effect in design' (Terzidis, 2006, p. 25).

Although the computer as a 'design tool' has made a great deal of progress since its limited beginnings as an analytical aid in engineering, there has always been a great deal of debate with regard to whether the computer can ever be more than merely a 'tool' in the design process. The notion of a computer being used to actually design is seen by some to be utterly

ridiculous, because design is seen as a human creative activity, impossible for a computer to recreate or mimic. The computer has thus, for the most part, been relegated to the position of a tool, carrying out tasks such as visualization (drafting, modelling and so on) and performance analysis.

However, algorithmic design approaches may change this view of computers. Recent studies suggest that it may be possible to bring human creativity and the computational power of computer design systems together by using algorithmic design approaches (Frazer, 1995; Kalay, 2004; Kolarevic, 2003; Kotnik, 2006; Oxman, 2008; Terzidis, 2006). The use of algorithmic design approaches could bring about a more true and real 'Computer Aided Architectural Design' (CAAD) than ever before. The use of algorithmic design approaches offers a great potential for bridging the gap between the human mind and the computer. Computer systems developed for use in the realm of design are endowed with a number of essential computational mechanisms, such as iteration, recursion and the conditional application of rules. Tiresome and time-consuming labour-intensive routines can be automated by algorithmic computational procedures, such as deduction, induction, abstraction, generalisation and structured logic. In addition, these algorithmic computational procedures are also found to be far superior to the human mind in terms of quantitative capability. Algorithms can even be designed with an inductive logic that can then produce unpredictable, unimaginable and unconceivable results. Similar to human intuition being seen as a source of human creativity, the inductive logic of an algorithm can be seen as a computer's creativity (Aranda & Lasch, 2006). As Terzidis (2006) points out, however, no matter how flexible and ingenious an algorithmic design system can be, computers are not able to consciously justify simple decisions, because they lack the abstract holistic nature of human thinking. It thus seems safe to say that the human designer is in no danger of being replaced by the computer.

Terzidis (2006) argues that one of the many differences between the traditional design approach and the algorithmic design approach is that, in the traditional design method, a 'top-down' strategy is followed. In

algorithmic design, on the other hand, a 'bottom-up' approach is taken. Although, with regards to the conventional design process, this argument might be correct in a general sense, it is not always the case, because the conventional traditional design process readily adopts many design strategies, including the bottom-up approach. In fact, many designers, such as Jiricna, have become famous for following a design approach that starts the design process by detailing the essential elements of the design through creating a grammar scheme in parallel to the spatial concept (Lawson, 1997).

The essential difference between the capabilities of the computer and the human lies in the nature of the information required by each and the processing of that information: that is to say, closed logic vs. open logic. Human architects 'deal with approximate knowledge and require an open logic system' while the computer normally 'operates on a closed logical system' (Flanagan, 2005). The computer can process many of the constraints that arise during the design process, because they are factual (environmental factors or site context); however, many other constraints may be less definite and more ambiguous, which means that they are nearly impossible to encode into a closed logic-based computer system. In addition, the designer's preconceptions of self-imposed constraints encompassed in the initial design idea are based on subjective, approximate factors, making this yet another set of requirements that are difficult to encode into the computer. All known constraints, whether factual and definite or ambiguous and subjective, must be represented and encoded in a manner that can be subsequently entered and understood by the computer system.

Most researchers, who support algorithmic computation in design, stress the need for a new design approach, which does not forcibly reflect the conventional design method, to be adopted. This design method should operate in the domain of computer science for it to benefit fully from the recognised potentials behind it (Frazer, 1995; Terzidis, 2006). Within this thesis it is argued that it is logical to recognise the change of the design

media from which the conventional design approach is developed into the digital design approach of the computer domain, it is not logical to fully abandon a design methodology that has proved its viability through years of practice and the production of many successful artefacts. Without forcibly adopting the conventional design approach in digital architectural design, the computer science domain is capable of fostering the domain of design while being adapted to incorporate issues, such as creativity and preconceptions, which are found to be crucially influential factors in design.

1.1.3 Parametric and modular design

The field of information systems has undergone many rapid advances in the last few decades and these advances have transformed the architectural, engineering and construction industries by incorporating a new perception of modularity. Computer applications developed for architectural design and manufacturing technology using algorithmic and parametric tools have revolutionised the production of the construction elements. Modularity is no longer regarded as a static term and design realisation is seen in a new light with the possibility of economic and efficient building design and construction coming to the fore. A variable set of parametric rules and mechanisms that allow for a wide variety of diverse and unique building components, all inherited from the same associative module, define the new module (Agkathidis, 2009). The designer can produce a new model by defining or declaring the parameters of a design. This new model will thus include variable geometrical definitions and behaviours. The parametric method of design allows the designer not only to define the relationship between the various design components of a parametrically defined model, but also to set the constraints or rules governing the defined variables. Thus, the model can be manipulated through the design process without losing control of the design itself (Shah & Mäntylä, 1995). The parameterised module, which incorporates technology, complex geometry and ornamental aesthetics, can, therefore, also achieve the desired efficiency and cost reductions (Agkathidis, 2009).

A standardised design approach to repeated projects is a controversial subject, because it conflicts with what is seen to be a vital architectural principle — that of seeing each building as a unique project. It is commonly agreed that the design decisions made during the earliest phase of the design process can significantly affect the overall construction and operational costs of the project (Gallaher, O'Connor, Dettbarn Jr & Gilday, 2004). Therefore, it is not difficult to see that the re-building of a successful design in a different context to that for which it was initially designed carries with it a significant potential for failure. The 'after-the-fact' element of trying to manipulate and 'fix' the areas where the building design might be failing to respond to the new context or environment could involve very high cost solutions.

Within this thesis, the parametric method of design is considered to present a new modular approach in design that has the potential of gaining the acceptance of designers who fought for their artistic freedom and rejected the idea of being 'assemblers of predesigned components'. The parametric design approach represents the new modular approach in design, because it provides designers with the freedom to create their own components and their own assembling techniques. Consequently, a new sense of standardisation can be approached. The building of constituent parts, together with the rules regulating their assembly, could be standardised, rather than standardising the whole building design every time a new building of the same kind is required to be built in a new location. The standardised building components could be reassembled according to and in response to the changing environments.

1.1.4 Integrated interdisciplinary design

The notion that architectural design should approach a building as a whole rather than as a collection of apparatus, and integrated between its external and internal orders, was brought to the forefront of architectural design discourse by Rush (1986). Rush, however, was not the first researcher to take an interest in this idea of integrated interdisciplinary architectural design; a number of researchers prior to Rush had already begun

researching the topic (Bachman, 2003). According to Bachman (2003), interdisciplinary integration is, and should be, a part of the architectural design philosophy. The integration of multiple disciplines into the design process is an evolving approach in architectural design. Integrated design is not a stylistic or comprehensive design approach; rather, it is an approach to design that is becoming progressively more crucial for producing functionally responsive, sustainable, economic design solutions.

Clearly a building designed on the basis of physical structure alone, without considering the environmental and social factors, would not satisfy, or even 'satisfice' (Simon, 1996), the vast majority of requirements. Instead, the design impetus should be driven by performance and system-based concerns. To effectively evaluate the performance and reflect back on the design decisions in relation to the simulation results, appropriate criteria, directly related to the design actions, must be selected. Although building performance tools can be used to optimise a few design parameters or to support focussed design decisions (mainly in the later stages of the design process), they cannot be seen as integrations in the building design. In an attempt to foster the integration in architectural designs, attempts have been made to integrate simulation tools earlier in the design process. Some of the most notable attempts include 'incremental levels of complexity', 'playing around with an idea', and 'simple generative form and genetic algorithms' (Souza & Knight, 2007). The methodological problems of setting up the performance evaluation criteria and subsequently relating them to the design actions are independent of the selected simulation tool.

During the early phases of the design process, building performance simulation is usually carried out on design solutions that have been produced. These performance simulations are undertaken as performance assessments that help to highlight potential areas for improvement and optimisation. Frequently, the results draw attention to a range of conflicting issues, constraints and requirements, and the designer is inevitably forced to make trade-off decisions. Unfortunately, these decisions are based on subjective design judgments, which is clearly not an

ideal way to find the most optimal design solution. A method for resolving this problem would be to produce design solutions in response to chosen performance criteria, rather than producing a design solution and then testing it against performance criteria.

Much research has been carried out on using evolutionary systems in architectural design to do exactly that (Caldas, 2008; Janssen, 2004) and, in many cases, it has successfully demonstrated that design solutions could, indeed, be evolved in response to performance criteria. However, in the vast majority of cases, the number of fitness functions assigned to the evaluation phase was limited and thus the interdisciplinary nature of the building design problem was not adequately reflected.

1.2 Outcomes of this research project

Much time and resources are required for all the stages ('conceptualisation', developmental' and 'evaluation') of this research to be fully implemented. This research project focuses mainly on the first two stages with a partial implementation of the third stage. The main outputs are as follows:

1. A meaningful research question and the hypothetical research proposition reflect the research goals and objectives.

2. A design method is based on the parametric and evolutionary approaches. The method adopts the design schema as the intermediate interpreter of the designer's creativity.

3. A computational architecture for the HEAD System describes the functionality and requirements of the system components and the interrelationship between them. The computational architecture of the system illustrates the system's design process and procedures.

4. A design of the database for the design schema produces the seeds of the graphical representations and the higher-level descriptions of buildings.

5. A Synthesis Algorithms with a multi-level hierarchical approach of a generative and an evolutionary design system and a multiple fitness function of interdisciplinary integrating design approach is developed.

6. The illustrative paper-based simulation tests the proposed design method and systems on a chosen project.

1.2.1 The proposed design method

The design of a building is a complex process that includes the search for solutions to a number of diverse needs. Architectural design and the search for an aesthetically-pleasing solution plays a large part in the design process but, even at the most basic level, various other disciplines within the building and construction field greatly affect both the ultimate design solution of the building and the design process itself. Any design problem raises a number of discrete sub-problems that are the primary focus of one or another of the various disciplines involved in the design and construction of a building. However, many (if not all) of the problems are found to overlap between disciplines and in many cases, the requirements and limitations from one discipline will often conflict with those from another. Thus, for an effective solution to be found, interaction, cooperation and coordination between the various disciplines is required from the very early stages of the design process.

The proposed design method acknowledges that there are a variety of discrete sub-problems and a range of requirements within each design problem. In an effort to manage the multiple tasks and requirements of each of the discrete design problems in a coordinated and cooperative way, the proposed design method takes a holistic approach to the overall design problem, taking into account all the different performance requirements of the building. While a holistic approach will tackle each discrete problem (sometimes resulting in a compromise between the various design requirements), it will normally produce a design solution that answers the

majority of the building's requirements as a whole, as satisfactorily as possible.

Application of the tectonic design approach (discussed in Section 3.2.3), to evolutionary design with its algorithmic imitation of the genetic propagation of species in response to their living environment, a multi-domain building performance simulation can be achieved. This multi-domain building performance simulation is expected to offer the designer the essential capabilities of Augenbroe's (2002) wish list, which would include, for instance, the ability to rapidly evaluate alternative designs, allow for the design process to be a rational decision-making process, allow for the overall design to be undertaken in incremental design strategies and allow for problem solving to be carried out in nonlinear, mixed and hybrid simulations.

The proposed design method demonstrates a means of digitally formulating the creativity of a human designer into a dynamic design schema that can subsequently be encoded and executed through the use of computational algorithms. Taking into consideration the tectonic design approach, which places a higher aesthetical value on the process rather than only on the product, the design schema is flexible and dynamic to be used appropriately for either a unique building design or for a standardised building design. The designer's creativity is translated into a modular design approach of parameterised building blocks. Prolog descriptions of the building's configurations and regulating elements are used to govern and guide the production of the design solutions throughout the permutation and propagation processes.

Genetic evolutionary algorithms hold a significant potential for the design process and the proposed design system exploits the many benefits offered by them. Not only does the proposed design method seek to apply genetic evolutionary algorithms as the main engine for actually generating design solutions, but it also seeks to use genetic evolutionary algorithms in the performance optimisation of the successful evolved models. The genetic

evolutionary algorithms can optimise the successfully-generated models in response to the simulated environment of the building in conjunction with both the internal and external constraints of the design problem.

1.2.2 The Hierarchical Evolutionary Algorithmic Design (HEAD) system

The HEAD system has been designed based on the proposed design method to comprise two main connected components. The first component is the Design Schema and the second is the Synthesis Algorithms. The Design Schema, includes the designer's personal input into the design system and is stored in a database. The Design Schema maintains design integrity and the designer's unique creative input into the design process. It also forms the basis for making decisions on any sub-problems that may arise from the second component of the system. The Synthesis Algorithms component is run by generative evaluation and optimisation algorithms and seeks to run the process of generating designs and the transition of evolved designs from one state to the next.

Performance-based design methods that are currently used in architectural design use analytical simulations of generated design solutions. The HEAD system takes a different approach: the design solution is actually generated and formed in response to the project's simulated environments. By generating design solutions in response to their simulated environments (rather than analysing solutions retrospectively), each design solution is unique in its environment. Design solutions that are unsuited to the specific design space are thus not generated, which reduces waste in terms of the time involved in generating design solutions that will quite clearly be rejected at a later stage in the process. An exhaustive process of generating a solution for each combination of design variants is unnecessary when a host of solutions can be ruled out early in the design process by generating the solutions in response to the simulated environment. Fewer alternative design solutions being produced has a carry-on effect, thus waste can be deleted from the optimisation cycle because fewer solutions need to be optimised. Another key difference between the current performance-based

design methods and the proposed design system is that the existing methods only allow for analytical simulations and subsequent modifications to be carried out on previously-generated design solutions. This HEAD system, on the other hand, will permit the automatic adjustment and development of the design solution during the actual generation and formation process.

There is one more aspect of the HEAD system that will aid the design process — this is the system interface, which is designed to ease and facilitate communications between the designer and the computer. The interface acts as a go-between between the 'real world' and the 'symbolic structure' of the system. The HEAD system interface will, in effect, allow the human designer to 'talk' to the system when he needs to supply or correct information that the system is processing. In addition to this, the HEAD system interface will offer the designer a means of entering his decisions into the system, when the system reaches a point where it cannot go forward with its available information.

1.2.2.1 Design Schema

The Design Schema component of the HEAD system is comprised of two sub-components, a database sub-component and a consistency-controller sub-component. The database sub-component is made up of every object and element contained in a building, in addition to the topology-geometry relationship between each of these components. The consistency-controller sub-component is designed to regulate the design process, the part-whole relationships, the design hierarchy, the scaffolding of the design process, the structuring of sub-problems and the structuring of the design parameters.

According to Picon, cited by Lim (2009), the designers traditionally perceived the world through the lens of the static laws of order and proportion, but this has changed to a view of the world in terms of the dynamics of change and movement. The classical geometrical design approach defined the beauty in the built environment in terms of the

designed object itself. More contemporary approaches, on the other hand, see the beauty in the dynamics of the design process. While the classical approach was based on geometrical realisations, contemporary approaches are based on mathematical and algorithmic processes of constructive patterns (Lim, 2009). This realisation that the dynamic processes of design themselves (and not just the designed product) can be inherently beautiful is a notion explored by tectonic realisation. In tectonics, the aesthetics of the process of the design and the organisation of the designed product are valued as highly as the aesthetics of the designed product itself. Contemporary technological design processes can thus be seen as 'process-oriented design', as opposed to the traditional 'object-oriented design'.

The 'design schema', as it is introduced in this research, is seen as the designer's project specific design model, framework or process. Because the schema is unique not only to the project, but also to the designer, it acts as a medium through which the designer is able to transfer his creative vision into the design solution. The design schema, as envisioned in this research, is the result of the designer's preconceptions (his/her guiding principles, primary generators and personal schema) interacting with the constraints and design problems of the specific design problem at hand. The design schema is seen as being dynamic and unique to each design problem. In addition, the design schema is also seen as a reflection of the designer's creativity. In the realm of computer-aided architectural design, encoding the design schema into a language that the computer can understand poses a challenge. The HEAD system acknowledges and respects the need for the design schema to be unique to each design problem and to each designer to retain the creativity of the designer in the design process. The HEAD system uses a variety of ways to encode the design schema into a language a computer can understand, while also maintaining this creativity and uniqueness.

1.2.2.2 Synthesis Algorithms

The Synthesis Algorithms component of the HEAD system has four hierarchical levels, room, layout, building and optimisation. At each level of

the hierarchy, goals are set and the solutions generated at each level are expected to achieve this set goal before being fed into the next level. Each level of the Synthesis Algorithms component is equipped with a generative transformational algorithm sub-component, an evolutionary genetic algorithm sub-component and an analysis, evaluation and selection sub-component. The designer can set the fitness functions against which each solution will be assessed. These fitness functions are based on the designer's interpretations of the design constraints and the simulation of the internal and external design environments. The genetic evolutionary algorithms use the fitness functions set by the designer and elaborate them into sets of targeted sub-goals.

The HEAD system will provide an integrated, dynamic design methodology. While the outputs, performance goals, optimisation cycles and controls can all be targeted in specific aspects of the design problem, the ultimate design solution produced is for the building as a whole, answering the overall design problem. By breaking-down the design problem into sub-problems and the building design task into sub-tasks, the Synthesis Algorithms component can more accurately assess and process information and find solutions for problems on a micro level. The Synthesis Algorithms component will process the problems at the first level before moving onto the next level equipped with solutions from the first level — this enables the system to deal with a far greater 'depth' of information, which could not have been contemplated if the design problem was looked at all at once and as a whole. The solutions from each level of the HEAD system are amalgamated, from one level to the next, into an overall design solution that is ultimately optimised at the final level of the HEADS.

The Synthesis Algorithms component is configured in a hierarchical, geometrical, multi-level manner. This set-up cuts-out the generation of unviable solutions and this, in turn, leads to a reduction in the time required for unnecessary computational processes. Because the hierarchical nature of the HEADS also allows for the design problem to be broken-down into smaller sub-problems means that each sub-problem and

solution can be assessed, using fitness functions, throughout its journey through the Synthesis Algorithms component. The assessment of each generated solution carries on throughout the design process and the ultimate 'overall' design solution produced is also assessed against the fitness functions. Throughout the evolutionary process, consistency controllers are used as a base and a reference for the design decisions that need to be made.

1.3 Overview of Thesis

The first chapter, 1 Introduction, is divided into the three main sections. The introductory chapter introduces the reader not only to the background of the topic under research, but also to the direction and aims of the research. The first section, Section 1.1 Overview of problem, introduces the background of the subject of this research. The subjects of creativity in design, algorithmic architectural design, parametric and modular design and integrated interdisciplinary design are all briefly discussed. The second, Section 1.2 Outcomes of this research project presents the main outcomes of the research and introduces the HEAD System. The third section of the introductory chapter is an Overview of Thesis (this section), which briefly outlines the structure of the thesis.

The second chapter of this thesis, 2 Research Design, is composed of four main sections, 2.1 Research objectives, 2.2 Research proposition, 2.3 Significance and potential benefits and 2.4 Research methodology. Section (2.1), outlines the objectives of the research project and lead into Section (2.2), which states the research proposition. Section (2.3) outlines the areas that the author feels have great potential significance. In this section, the hoped-for benefits resulting from the research are highlighted. Section (2.4), outlines the research methodology undertaken and explains the reasoning behind integrating various aspects of the different methodologies into a multi-methodological approach.

The third chapter of the thesis, 3 Literature Review, consists of three main sections, 3.1 Theoretical overview, 3.2 Computer Aided Architectural Design (CAAD) and 3.3 Space layout planning. Each section consists of a review and discussion of the previous research into the main areas of interest for this paper. The Theoretical overview, Section (3.1), takes a high level view to outline some of the most significant historical aspects of design. Current knowledge and opinions regarding design in general and architectural design in particular are also reviewed in this section. Section (3.2) is a review of the research into the use of computers in the architectural design process. The distinction between computation and computerisation is highlighted and the fact that most existing CAAD systems represent 'computerisation' is identified. Section 3.3 Space layout planning is a review of one of the critical aspects of architectural design. In this section, current methods of computer-aided space layout planning are discussed and the notion of using graph theory to facilitate the planning of the spaces within a building is explored. Section 3.4 Generative and Evolutionary design systems is a review of the current research into computer systems, based on the use of algorithmic operations, which can be used to actually generate forms and evolve those forms into new forms.

Chapter 4 Conceptual Design Method, provides an in-depth study of a way in which the current knowledge can be amalgamated into a workable computer-aided design method. Frequently, there is a disconnect between the various schools of thought, research and existing methods within the domain of building design; this section seeks to bridge the gaps and use the benefits offered by a variety of research findings. The fourth chapter is divided into six main sections, 4.1 The Reference and Impact Model, 4.3 Introduction, 4.4 Design context, 4.5 Designers' preconceptions, 4.6 Design schema, 4.7 Encoding, and 4.8 Generative evolutionary architecture.

Section (4.1), highlights the established Key Factors and the Measurable Successful Criteria that the in-depth reviews of the relevant areas were further researched. The introductory section (4.3), briefly outlines the conceptual design method and provides a graphical representation of the

various aspects of the design method highlighting the inter-connected relationships between each part. Section (4.4), explores the Design context, which includes everything that affects or is affected by the design problem. There are numerous aspects that make-up the design context and these are discussed in relation to how the designer and the design method can best deal with the wide variety of often-conflicting considerations that need to be solved.

Section (4.5), looks at the notion that no designer approaches a problem as a blank slate. Each design problem is tackled by a designer bringing a set of preconceived notions, beliefs and indeed knowledge and experience to the problem. The way in which this affects the eventual design solution and how this subjective, creative input can be maintained in a CAAD method is discussed. Section (4.6), introduces a new conception of the schema. Current thinking sees the schema as the amalgamated experiences of the designer that contribute to a designer's 'style' that is applied to each design he/she approaches. In this section, the notion is explored that the schema is not only specific to a designer but also becomes specific to each design problem.

Section (4.7), examines ways in which the open logic of human thinking and creativity can be translated into the closed logic system of computers. For any CAAD method and system to be successfully implemented and used, the gap between the computational systems used and the human mind must be bridged. Section (4.8), explores the use of nature's evolutionary processes in architectural design.

Chapter 5 Demonstration and Illustration, consists of two sections; (5.1 HEAD System Architecture) and (5.2 Illustrative paper-based simulation: Mosque). The first section (5.1) is a detailed discussion of how the proposed design system is to be set up and provides a software system architecture for the eventual implementation of the proposed system. The second section (5.2) covers the illustrative paper-based simulation that was

undertaken to prove the validity of the proposed design method and system.

Chapter 6 Discussions and Conclusions discusses the research undertaken and the conclusions drawn from the results obtained.

2 Research Design

2.1 Research objectives

The design concept is conceived at the very early stages of the design process. This conceptual stage produces the 'conceptual design' that is evaluated and explored before the design process develops any further. The process of conceptualising the design idea is not as straight-forward as one would imagine, because the information regarding the design problem at hand is frequently incomplete and imprecise.

Although these early stages are the least resource intensive, the cost implications (for the entire project) of making sub-optimal choices at these early stages means that any improvements to the conceptual design will have a great and lasting impact on the entire design process. The overarching aim of this research project is to find and develop a system that can support the designers at the early conceptual design stage with the ultimate goal of providing the designers with the capability of producing functional, high performance, sustainable, aesthetically pleasing and responsive designs.

The 2004 study, 'Cost Analysis of Inadequate Interoperability in the U.S. Capital Facilities Industry', conducted by the National Institute of Standards and Technology (NIST), U.S. Department of Commerce, estimates that at least US$15.8 Billion is lost annually because of the lack of the ability to share, exchange and manage information between the project stakeholders (Gallaher et al., 2004). While acknowledging the building industry's shift toward an integrated collaborative practice, it is realized that architects, as designers and collaborators and with the impact of the design solutions they provide during the early phases of the project, have the potential for playing a much greater role within the building process as a whole. (Figure 2-1 MacLeamy's Curve), derived from the field of cost engineering, shows the 'traditional design process' effort curve (3) where the "ability to impact cost and functional capabilities' decreases while the 'cost of design changes'

increases. Most effort is expended when the ability to control cost is minimal and the cost of change is thus higher. It also separates the designers from the contractors. Collaboration between all the project teams from the early phases of the project will shift the effort curve (4) to where the design decisions are more effective and less expensive.

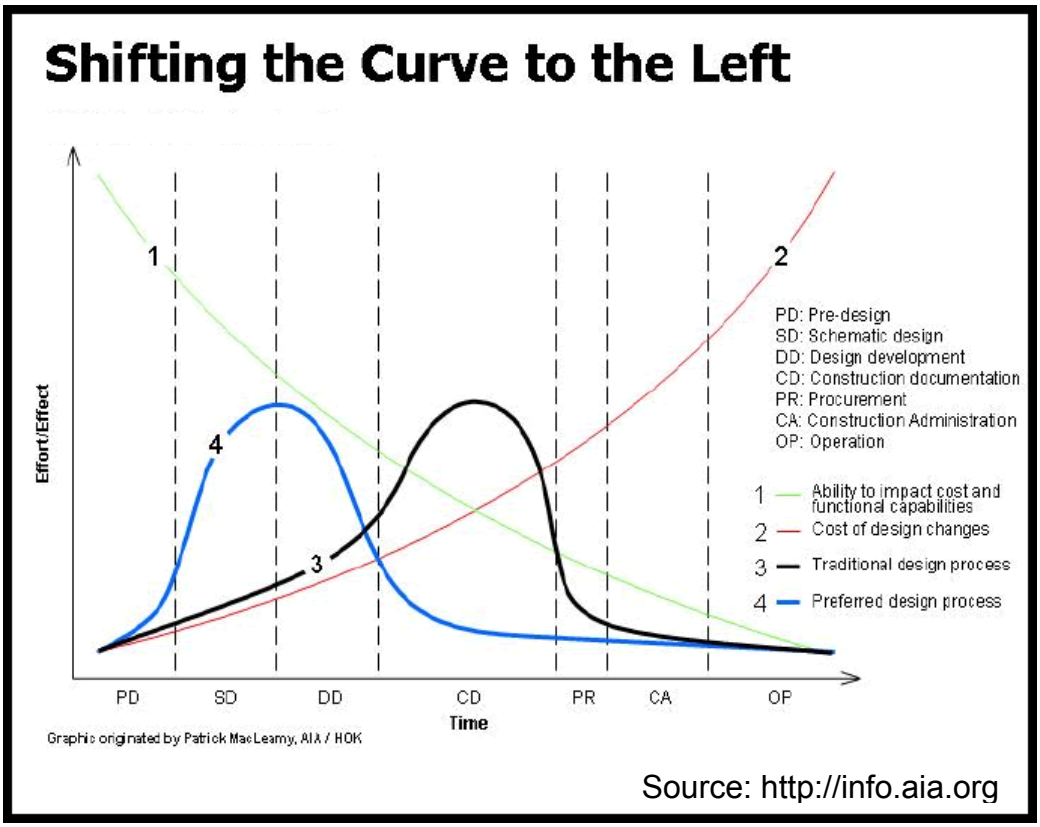


Figure 2-1 MacLeamy's Curve

As the proposed design method takes an interoperable approach, it falls in-line with the industry practitioners' intentions of shifting the effort to the earliest phase in the project. The project design constraints themselves reflect the different stakeholder requirements that influence the design (as discussed in section 4.4.1). The different disciplines involved in the design process i.e. the design team, are involved in a more collaborative way during the early stages of the design process that means that a higher level of interoperable communication is facilitated. Ultimately this will lead to the overarching goal of cost reduction. Furthermore the time, cost and risk a designer has to invest in a given project will be greatly reduced. In

addition, the quality and accuracy of the final design will be greatly enhanced by the computer applications.

At the General Directorate of Military Works (GDMW), the building procurement arm of the Saudi Arabian Armed Forces, those arguing in favour of standardisation make the point that standardisation of less important projects would allow for more time and effort to be directed towards selected, more important and more complicated projects. However, this is still controversial, because standardisation would mean that buildings are not designed in response to their specific environment and needs, thus potentially resulting in sub-performant buildings. This highlights the need for a design method that incorporates certain standardised elements into the design process while, at the same time, maintaining the vital role that both the requirements of each specific design problem and the designer's human 'subjective' creative vision play in developing a functional, high performance, sustainable, aesthetically pleasing and responsive design solution. The ultimate aim is develop a design method and system that will be able to evolve designs in response to their living environment and the requirements of each discrete design. The notion is to be able to produce the same building, or same type of building, and thus substantially reduce the design time and costs in any number of locations and yet still be able to produce unique designs in each location that evolve to suit the specific location and needs.

It is expected that the design method developed in this research would be highly appropriate for use in a design processes that can be standardised, for instance, in the design and development of specific building types that share common design requirements. However, it is believed that that the design method could also be equally applicable and appropriate in the design of any design process. Whether or not the design process is standardised, it is obvious that there is a great potential for developing a design method capable of producing design solutions that are unique and vary according to the specific environment of the project. It is believed that a design system that can offer a balanced alternative to the extremes of

complete standardisation and completely non-standardised design processes can be developed. In effect, the goal is to achieve a middle ground that can take advantage of the time-and-cost savings of standardisation, while also producing unique, responsive and performant buildings.

More explicitly, the goal of this research is to develop a computer-aided design method that will support conceptual design through an integrated interdisciplinary design system.

2.1.1 Research question

How can generative and evolutionary design promote an integrated, interdisciplinary building design approach to support multi-evolutional criteria while fostering a collaborative relationship between human creativity and computer capabilities at multiple levels of detail?

2.2 Research proposition

The research hypothetical proposition is that an integrated interdisciplinary design approach that takes advantage of the generative design system and evolutionary genetic algorithms **could** be used during the early stage of the design process. A design system is envisaged that connects the evolutionary systems directly to the performance assessments applications as fitness functions organised in order of priority. The notion is to develop a design system that facilitates the smooth transition from the designer's creativity into the computer system. This would be most efficient if the pre-geometric aspects of the design were used as inputs directly into the evolutionary systems.

2.3 Significance and potential benefits

The intention of the overall research project is to conceptualise and design a computational design method and system that has the potential for being used from the very earliest stages in the design process. The intention is to design a system that will approach and respond to those issues within the design process that are believed to be of great importance, such as the

subjectivity and objectivity of the designer and the design team, economic considerations and building performance.

Taking a high-level broad overview of the research project, some key benefits that could be gained from the research can be identified. First, because the aim is to design a design system that can be used from the very earliest phases of the design process, it is believed that the closed-logic of the computer system can aid the designer in making more objective decisions. The important benefit here is that although a certain amount of objectivity could be injected into the decision-making process, the designer could, potentially, still be able to maintain certain advantageous aspects of his subjective creativity. Second, by enabling the designer to make more objective, justified and sound decisions at the earliest stages of the design process, the envisaged design method and system would help to minimise the most costly and disruptive changes that might occur later on in the project. The third potential benefit of the overall research project is the fact that the design method and system that is being researched have the potential for promoting an integrated, inter-disciplinary design process that would advance greater collaboration between not only the design team but also the project stakeholders. This represents another cost saving in terms of dealing with potential conflicts between the various disciplines. In addition it is believed that the inter-disciplinary nature of the design method and system could actually produce better design solutions that respond to the functional, aesthetic, social and economic requirements in a more balanced way. The fourth and final major benefit that could result from this research project is that the design solutions produced by the envisaged design system would be performant and sustainable, because the models/solutions would be evolved in response to their environments, both external and internal.

Form the perspective of CAD as a whole, this research holds the potential for bringing the human designer closer to the computational powers of the computer. Current research and CAD systems offer designers a ‘computerisation’ of the design process that can aid primarily in the

visualisation and production of the design. The goal of this research is to enable the designer to share a symbiotic relationship with the 'computational' aspects of the computer through the use of the genetic evolutionary algorithm. Although the genetic evolutionary algorithms that the research is based on would ordinarily require a high level of expertise in programming, it is believed that, by injecting some borrowed aspects of the traditional design process, the level of the programming knowledge required of a designer can be minimised.

2.4 Research methodology

2.4.1 Methodological pluralism

Blessing and Chakrabarti (2009) highlight that design research has undergone a number of phases that have overlapped as they progress from one to the other. These phases are the Experiential, the Intellectual, and the Experimental. Until the late 1950s, the experiential phase predominated. Senior experienced designers, who wrote about their experience, observations and work within the design process, carried out experiential design research and, of course, the resulting design products. Experiential design research did not, however, have any theoretical framework into which the observations and experience of these designers could be placed and each designer's experiential research generally tended to be very specific to their particular domain. By the 1960s, the Intellectual phase in design research began and lasted for approximately 20 years. Intellectual design research placed a far greater emphasis on logical and consistent design. This phase was characterised by the development of a wide variety of design methodologies, principles and methods. The 1980s saw the beginning of the Empirical phase of design research, which had gained more interest and drive by the 1990s. The empirical approach to design research seeks to gain a greater understanding of *how* designers and design teams actually *design*. The empirical approach also seeks to understand the affect of various design methods and tools on the design process (Blessing & Chakrabarti, 2009).

Methodological pluralism, which can be seen as a combination of intellectual design research and empirical design research, calls for multiple theoretical models in addition to multiple methodological approaches in scientific practice (Norgaard, 1989; Polkinghorne, 1983). Polkinghorne (1983) argues 'that human science requires a syncretic approach which integrates the results obtained through multi-schematic and multi-paradigmatic systems of inquiry' (Polkinghorne, 1983, p. xi). Similarly, Howard (1983) 'believe[s] that a thorough understanding of humans will be facilitated by methodological pluralism' (p. 20). Nunamaker, Chen and Purdin (1990, p. 91) appear to agree with the need for multiple research methodologies when they point out that while 'some research domains are sufficiently narrow that they allow the use of only limited methodologies ... [other] research domains are sufficiently broad that they embrace a wide range of methodologies' (Nunamaker Jr et al., 1990, p. 91).

2.4.2 Research approach

The overall research project is developed by combining the two approaches presented by Nunamaker Jr and his colleagues (Nunamaker Jr & Chen, 1990; Nunamaker Jr et al., 1990) and Blessing and Chakrabarti (2009). Nunamaker Jr and his colleagues (Nunamaker Jr & Chen, 1990; Nunamaker Jr et al., 1990) presented what is, in effect, a multi-methodological approach to Information System Research. Blessing and Chakrabarti (2009) also support the need for multiple methodologies in research and present *'A Design Research Methodology'* which combines a mixture of types of research. Both methodologies recognize particular research methods, prescriptive and descriptive, where each one is applicable during a specific stage of research.

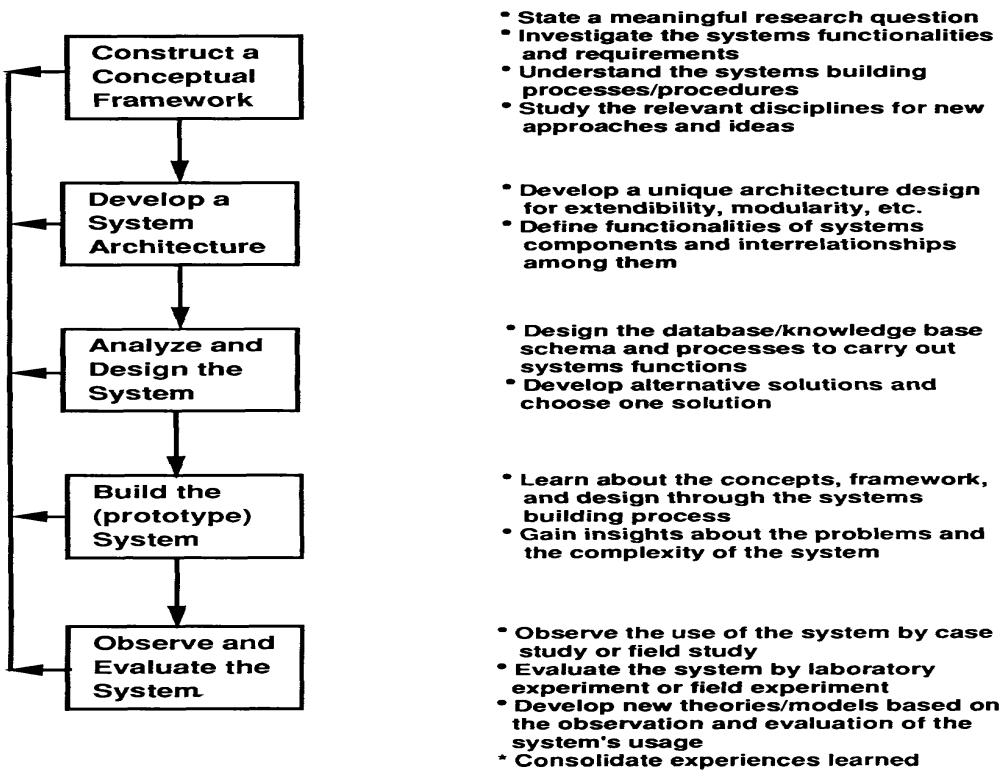


Figure 2-2 Nunamaker's (1990) research process

Frameworks in the information system development research, such as the multi-methodological research framework presented by Nunamaker Jr et al. (Nunamaker Jr & Chen, 1990; Nunamaker Jr et al., 1990), usually consist of three main stages, the 'conceptualization' stage, and the 'implementation' stage, which both usually use the prescriptive methods of research, and the 'evaluation' stage, which is a more descriptive method of research. Each of these stages can be broken down into smaller stages. The first stage of 'conceptualization' consists of the formulation of the 'user methods, theoretical models and system architectures'. Although the 'conceptualization' stage does not required demonstration, the theoretical models, methods and system architectures may be tested to verify their feasibility to be implemented in the next stage. The second stage of 'implementation' is characterised not only by what Janssen refers to as the 'proof-of-concept' (Janssen, 2004, p. 26) prototype, but also the development of entirely expressed production systems together with a detailed specification for the implementation. Testing the produced system

for performance, carrying out case and field studies and so forth occur in the third stage of 'evaluation'.

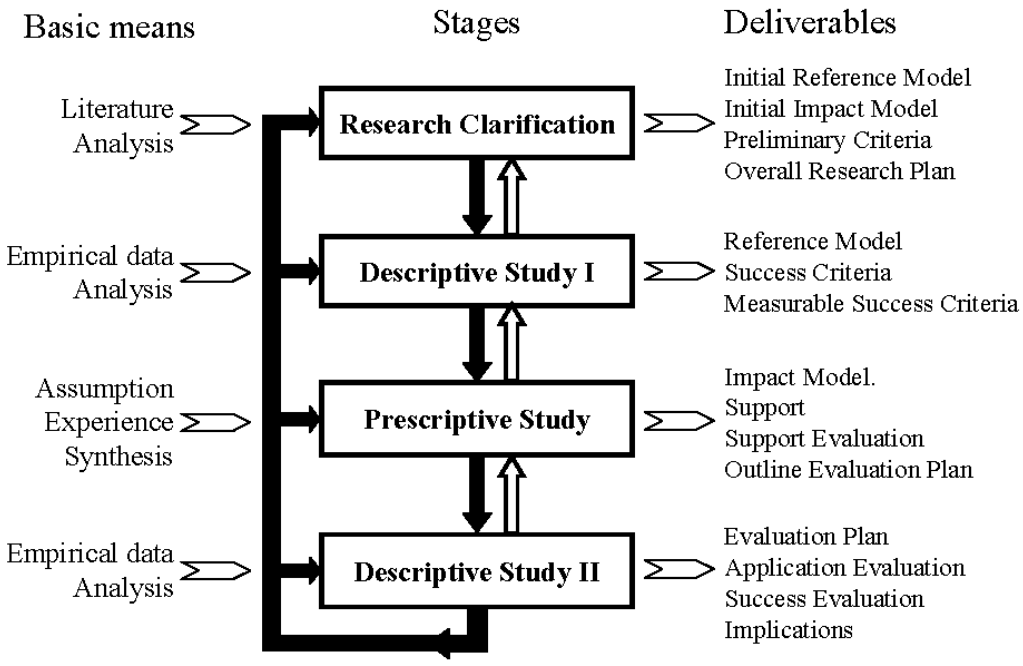


Figure 2-3 Blessing's (2009) DRM framework

Blessing and Chakrabarti's (2009) multi-disciplinary framework is divided into the four main stages of Research Clarification, Understanding Design, Developing Design Support and Evaluating Design Support. The 'Research Clarification' stage aims to establish a realistic research goal by searching the literature for substantiations that support the initial hypothesis. The expected 'deliverables' of the 'research clarification' stage are, 'an Initial Reference Model, an Initial Impact Model, a preliminary set of Criteria, and an Overall Research Plan' (Blessing & Chakrabarti, 2009, p. 41). The 'Understanding Design' stage is intended to descriptively detail the existing situation of the domain under research to determine those factors that should be taken in hand to develop a competent clarification of the task. The resulting deliverables of the understanding design stage are 'a Reference Model, an updated Impact Model, Success and Measurable Success Criteria, as well as implications of the findings for the development of support' (Blessing & Chakrabarti, 2009, p. 41). The 'Developing Design Support' stage uses the interrelated influencing factors that were

determined in the previous two stages, together with the researcher's experience and understanding, to develop a prescription of the intended design support. The deliverables of the 'developing design support' stage are expected to be 'an Impact Model, descriptions of the Intended and Actual Support, the Actual Support, Support Evaluation results, and an Outline Evaluation Plan' (Blessing & Chakrabarti, 2009, p. 41). The 'evaluating design support' stage includes two empirical studies for the evaluation of, first, the applicability, and second, the usefulness of the design support. As a result of the evaluating design support stage, 'Application and Success Evaluation results and suggestions for improvement of the Actual and Intended Supports, as well as the Reference and Impact Models' are expected as deliverables (Blessing & Chakrabarti, 2009, p. 41).

2.4.3 Research framework

The overall research project is developed by combining the four major stages of Blessing's (2009) Design Research Methodology (Figure 2-3 Blessing's (2009) DRM framework), with the five research stages of Nunamaker et al.'s (1990) research process (Figure 2-2 Nunamaker's (1990) research process). Blessing's 'Research Clarification' and 'Understanding Design' stages are similar to Nunamaker et al.'s 'Construction of a Conceptual Framework' stage. The 'Developing Design Support' stage of Blessing's Design Research Methodology is similar to both the 'Development of a System Architecture' and the 'Analysis and Design of the System' stages of Nunamaker et al. Blessing's 'Evaluating Design Support' is similar to the divided stages of the 'Building of the (Prototype) System' stage and the 'Observation and Evaluation of the System' stage of Nunamaker et al. As a result, the three main stages are distinguished here, the 'Conceptualisation', the 'Developmental' and the 'Evaluation'. Individual research projects can concentrate on any one of these stages and the output and results from an earlier research project can be used as the base for a subsequent research project. It is worth mentioning here that iterations

between the three stages of the overall research project are permitted, depending on the understanding of the design situation under research.

The three stages of the overall research project are discussed below in more detail highlighting the goals and the outcomes of each stage;

2.4.3.1 Stage I Conceptualisation

2.4.3.1.1 Research clarification

The first stage of the overall research project is the Conceptualisation stage that includes the research clarification, where a literature review of the areas of relevance and contribution are analysed to determine the goals and objectives of the research.

- Goals
  - Identify the overall topics of interest;
  - Clarifying the current understanding and expectation;
  - Clarifying criteria, main questions and hypothesis;
  - Determining areas of relevance and contribution;
  - Formulating research plan.

- Outcomes
  - Current understanding and expectation;
    - Initial reference model;
    - Initial impact model;
    - Preliminary criteria.
  - Overall research plan
    - Research focus and goals;
    - Research problem, main research question and hypothesis;
    - Relevant area to be consulted;
    - Research approach;
    - Expected area of contribution and deliverables;
    - Time schedule.

2.4.3.1.2 Exploratory review

Based on the findings of the research clarification, an exploratory review, which aims to reach a general understanding of the design method, is conducted. This exploratory review is based on a descriptive methodological approach. The resulting conceptual design method describes the general outlines and characteristics of the proposed design method and the proposed design system.

- Goals
  - Identifying factors influencing the preliminary criteria;
  - Complete the reference and impact model including success criteria and measurable success criteria;
  - Establish key factors that would lead improvement;
  - Use of understanding for the development of the intended system.

- Outcomes
  - Reference model;
  - Impact model;
  - Success and measurable success criteria;
  - Key factors;
  - Description of the intended system.

2.4.3.2 Stage II Development

The second stage of the overall research project is the Developmental stage that is based on a prescriptive approach, because it deals with the synthesis of the design method. The design system architecture is developed, based on the findings of the previous stages. The system components, functionalities, structural relationship and dynamic interactions among system components, all specify and present a road-map for the system building process. In addition, to analyse the system's data-structure, databases and knowledge bases, a specific standardised building is selected as an illustrative paper-based simulation with key aspects highlighted to demonstrate the feasibility of the proposed design system architecture.

2.4.3.2.1 System architecture

- Goals
  - Develop the architecture of the intended system;
  - Develop an outline evaluation plan for stage III.

- Outcomes
  - System architecture;
  - Documentation of the system architecture;
    - System architecture description;
    - Introduction plan;
    - Impact model.

2.4.3.2.2 Paper-base illustrative simulation

- Goals
  - Evaluate the developed system architecture to determine whether to proceed to next stage or not;

- Outcomes
  - Result of the system architecture initial evaluation;
  - Outline evaluation plan.

2.4.3.3 Stage III Evaluation

The third stage of the overall research project is the Evaluation stage, where a prototype system, which demonstrates the feasibility and the usability of the system, is built. The implementation of the design system by building the system prototype enables observing, testing and evaluating the system performance based on the criteria defined in the earlier stages.

2.4.3.3.1 System prototype

- Goals
  - develop the intended system prototype, highlighting the Key Factors against the Measurable Success Criteria;
  - evaluate the system prototype with respect to its in-built functionality, consistency, etc.;

- Outcomes
  - Documentation of the system prototype;
    - Prototype system description;
    - Prototype system introduction plan;
    - Prototype system impact model.
  - Results from the system prototype evaluation.

2.4.3.3.2 Evaluation

- Goals
  - identify whether the system can be used for the task for which it is intended and has the expected effect on the Key Factors (Application Evaluation);
  - to identify whether the system indeed contributes to success (Success Evaluation), i.e., whether the expected impact, as represented in the Impact Model, has been realised;
  - to identify necessary improvements to the concept, elaboration, realisation, introduction and context of the system;
  - to evaluate the assumptions behind the current situation represented in the Reference Model, and the desired situation represented in the Impact Model.

- Outcomes
  - results of the Application Evaluation;
  - results of the Success Evaluation;
  - implications and suggestions for improvement for:
    - the Actual Support;
    - the Intended Support, its concept, elaboration and underlying assumptions;
    - the Actual and Intended Introduction Plan including introduction, installation, customisation, use and maintenance issues;
    - the Actual and Intended Impact Model;
    - the Reference Model;
    - the criteria used.

2.4.4 Scope of this thesis

Multi-disciplinary and multi-methodological research frameworks incorporate extended research work that can only be covered by a range of research projects, with each research project concentrating on a specific type of research to achieve an overall targeted goal (Nunamaker Jr et al., 1990). To undertake each stage of a research framework, whether multi-disciplinary, multi-methodological, or indeed both, completely and thoroughly for each project would be impossible. Not only is it more than likely that the resources for such a complete undertaking would not be available, but it is equally likely that to cover each stage thoroughly would be a redundant activity. Depending on the research framework and the domain of research, the research activities can be broken down in different ways between the research projects (Blessing & Chakrabarti, 2009).

The individual research projects that make up the multi-disciplinary and multi-methodological research framework can be set up to concentrate on just one stage of research and can follow just one method that is deemed the most useful to that stage and domain of research. The discrete research projects, which together make-up the overall research framework, will clearly be interrelated at a variety of levels and in different ways. Multi-disciplinary and multi-methodological research frameworks are complex undertakings that can run over a long period of time. Given sufficient time, the results and conclusions from each individual research project, conducted at the different stages and using different methods, can be collated and integrated to eventually develop an overall result(s) to the question(s) posed by the research framework as a whole.

Although the individual research projects can be carried out as discrete projects, the results of one project may be used as the basis for beginning another research project or it may be found that the results of one research project may require that another project be re-visited at an earlier stage. The research projects that target the earlier stages of the overall research framework will produce results that are not usually useful for the

evaluation purposes. However, it is possible that different validation methods will have to be used for early stage research projects. Nunamaker et al. (1990), in their study of systems development in information systems research, find that for research projects in the applied sciences (for example, engineering, developmental and formulative research categories), 'proof' can be 'taken to be any convincing argument in support of a worthwhile hypothesis' (p. 91). Janssen (2004), citing Nunamaker et al. (1990), goes further and highlights the argument that it is very acceptable to evidence and support this 'proof' through the 'development of a method or system' (p. 27). This would effectively negate the need for research to produce results full of data, because the development of a system or method can 'be thought of as a 'proof-by-demonstration' (Janssen, 2004, p. 27).

For the evaluation of the system as a whole, it is usually the research projects tackling the later stages of the research framework that can be the most useful and applicable. A number of researchers, such as March and Smith (March & Smith, 1995) and Järvinen (Järvinen, 2000a, 2000b) believe that, regardless of which stage of the research framework a project concentrates on and whether it be conceptualisation, implementation or evaluation, its results and findings should be appraised from the standpoint of usability and utility to the community that will be affected by it. However, Nunamaker et al. (1990) point out that a typical research life-cycle takes the form of research into conceptualisation, development and then impact. The concept stage is characterised by creativity, the development stage is characterised by experimentation and research into implementation, and the impact stage is characterised by usability and acceptance testing. Thus, the evaluation of usability and utility to the community, that March and Smith (March & Smith, 1995) and Järvinen (Järvinen, 2000a, 2000b) call for would, according to Nunamaker et al.'s (1990) notion of the life-cycle of a research project, usually occur during the latter stages of the research.

For this kind of appraisal to be made, the results alone are not sufficient. Instead, the results from each research project should be judged by reasoning with reference to the existing context and environment into which the results of the research project and framework are intended to be used. When the results of a research project are judged to be original and important in some respect, it can be considered as research. The evaluation of the performance and implementation of the project is unnecessary for proving that the research makes a contribution. It is the originality and newness of the results that are considered as the 'contribution' of the research.

This thesis focuses on the first two stages of the overall research project of the Conceptualisation stage, the Developmental stage and the Evaluation stage. The first Conceptualisation stage determines the project's goals and objectives by stating the research question and the research hypothesis. A conceptual design method is developed, outlining the characteristics of the proposed design system. The second Developmental stage produces a detailed system architecture specifying the components, functionalities, structural relationship and dynamic interactions among the system components. At the end of the Developmental stage, to illustrate the theoretical feasibility of the system architecture a paper-based simulation, with the key aspects being highlighted is conducted. While relevant findings drawn from the previous literature are used in support of the system, the paper-base simulation is used to verify the design system architecture by illustrating the key aspects that have emerged as a result of this research. The third 'Evaluation' stage is not a part of the scope of the work in this research.

3 Literature Review

This Literature Review chapter reflects the main outcomes of the research Clarification part of the Conceptualisation stage of the overall research project and this thesis. The research Clarification is review-based research to identify the overall topics of interest and to clarify the current understanding and expectation and subsequently determining the areas of relevance and contribution.

In the first section of this chapter (Section, 3.1 Theoretical overview), the topics covered provide a general impression of the environment within which the research area exists. The review of the literature relating to the background subjects of design thinking and creativity, design, Design Methods and the role of the designer, are not, in any way, an attempt at a comprehensive, in-depth examination of these vast areas of study; each is worthy of extensive research on their own. It is beyond the scope of this research and the intention here is to give the reader a basis from which to tackle the main subject of this research.

The second section of the Literature Review (3.2 Computer Aided Architectural Design (CAAD)), briefly explores the research surrounding the use of computers in design. The distinction between computerisation and computation is discussed in Section (3.2.1 Computation vs. Computerisation), in Section (3.2.2 Digital architectural design methods) are reviewed and Section (3.2.3 explores the notion of Tectonic Design).

The third section of the Literature Review, (3.3 Space layout planning) seeks to cover the current literature on a central aspect of architectural design. The planning of the configuration of the spaces that make-up the building as a whole impacts every aspect of the design, including the functional, the aesthetic and the economic considerations. Research into the various ways in which the computational powers of computer systems can be used to aid in the process of space layout planning are examined. This section lays the foundations for a discussion on the potential ways in

which space layout planning can be further facilitated through the use of graph theory.

The fourth and final section of this chapter (3.4 Generative and Evolutionary design systems), begins with a review of the existing literature on algorithms (specifically, genetic algorithms that are a kind of evolutionary algorithm), that form the main 'processing power' behind generative and evolutionary design systems. Research into generative design systems is then reviewed. The aim of these generative systems is to use the computational powers of computer systems to search very large design spaces and to have the computer actually generate solution forms. Finally, there is a discussion on evolutionary design systems that are based on nature's evolutionary processes. Given 'parent' forms, the evolutionary systems, using genetic algorithms, are able to compute vast numbers of possible mutations and eventually produce a second generation of forms.

3.1 Theoretical overview

3.1.1 Design thinking and creativity

Creativity is generally regarded as being within the realm of unconscious thought. The different sides of the human brain function differently. The left part of the brain is logical and analytical and controls speaking, reading, writing, numerical operations and generally processes information on a conscious level. The right part of the brain, on the other hand, controls subconscious thinking. For instance, the right side dominates in spatial recognition, holistic understanding, sensing, visualization and intuition. The notion that the creative process is carried out by the unconscious, right side of the brain has meant that it is often seen as an inexact, hard to define and often irrational process. However, creative problem-solving, as in the case of design thinking is an incredibly complicated process which encompasses both conscious and unconscious though processes and there is no evidence proving that it is controlled and processed by just the right side of the brain.

Commonly seen as some kind of mysterious, God-given, special talent that only a selected few possess, creativity is all around us; it plays a part in all domains and disciplines in a range of forms and to varying degrees. The obscurity and enormous breadth of creativity has meant that, to use the words of Taylor, 'definitions of creativity are often misleading: they say too much and too little' (2009, p. 2). Over the years, scientists have attempted to formalise the concept of creativity into a more understandable, methodical process.

In the *Art of Thought,* first published in 1926, Wallas began the more scientific exploration of creativity (1949). He proposed a model of the creative process made-up of the four stages. The problem is first 'investigated in all directions' during the *preparation* stage. This is followed by an *incubation* stage where the problem is not consciously considered. The third stage of *illumination* is when the 'happy idea' or solution occurs. The fourth and final stage of *verification* is a conscious working out of the problem and the solution (Wallas, 1949). While accepting Wallas' creative phases, Taylor (2009) contends that there are actually a number of different levels of creativity. He placed expressive creativity, like that of children drawing spontaneously and with a disregard for the final product, on the first level. The second level, productive or technical creativity, occurs when the spontaneity inherent in the first phase is restricted and creative techniques are developed. The inventive level of creativity occurs when technical contributions are used to solve practical problems and ability is clearly visible in materials, methods, techniques and results. At the level of innovative creativity, skills in conceptualizing allow for improvements and modifications. Taylor's final level of creativity, 'emergentive' creativity involves entirely new principles or assumptions. According to Taylor, this final level is what is generally regarded as creativity. However, the four lower levels cannot be dismissed (Taylor, 1959). In fact, as Torrance (1995) pointed out, most of the investigations into creativity are carried out on the lower levels, as the highest level is indeed rare and a talent that only a few possess.

As Sternberg (1999) pointed out, Guilford had an enormous impact on the field of creativity when he distinguished between convergent and divergent production. Convergent production involves deriving the single correct answer to a clearly defined question, while divergent production involves the ability to draw on ideas from a broad search for information and the subsequent generation of numerous novel solutions to problems. In other words, divergent production is the creative generation of multiple answers to a set problem (Sternberg, 1999).

In *The Use of Lateral Thinking* De Bono (1971) proposes the theory that it is possible to produce new ideas artificially at will, rather than relying on inspiration. The successive data received through experience, according to De Bono, is not recorded by the mind in an objective way. Rather, experiences contribute to patterns that humans use for everyday existence. However, living and thinking according to these patterns means that new ideas are not developed unless you learn to look at things in a different way. This alternate way of looking at things is what De Bono called 'lateral thinking' (De Bono, 1971).

Wertheimer (1959) looks at productive thinking through a gestalt lens; that is, the whole is greater than the sum of its parts. According to Wertheimer, productive thinking is the ability to see a problem situation as a whole. When a problematic situation is first identified, it provokes first thought and then action leading to the restructuring of the situation. With further consideration, thought and restructuring, a satisfactory situation is ultimately reached (Wertheimer, 1959).

Ferguson (1977; 1992) in his work on the nature of engineering, explores the nature and significance of nonverbal thought. He proposes that much of creative thought is non-verbal, processed by the brain as visual images rather than words. The inability to reduce these visual, non-scientific thoughts and ideas into words or numbers precludes the production of novel solutions using systematic methods alone. Ferguson contends that a

systematic approach may lead to the atrophy of the nonverbal imagination (Ferguson, 1977; 1992).

In his paper *Discovery, invention, and development: Human creative thinking,* Simon (1983) confirmed Ferguson's contention that the creative process cannot be fully understood or explained by science. 'Much of what we are thinking is inaccessible to our conscious awareness' (Simon, 1983, p. 4569). On the other hand, Simon also points out that a great deal of the problem-solving achieved by computer programs would be regarded as creative if performed by a human. Using the Logic Theorist and BACON computer programs, Simon et al. proved that a computer program can make scientific 'discoveries' just as the human can, if the right heuristics and data are fed to the program (Newell, Shaw & Simon, 1958; Simon, 1988). As Liu points out, however, creativity can be replicated by a computer system 'if and only if some initial data and heuristic function towards creativity are available' (Liu, 2000, p. 275).

Csikszentmihalyi disagrees with Simon's notion of creativity being a problem-solving behavior (Csikszentmihalyi & Getzels, 1971). Citing Kuhn, Csikszentmihalyi claims that creative achievement can be seen in finding solutions to problems 'which were not even formulated as such, but had first to be identified as problems' (Csikszentmihalyi & Getzels, 1971, p. 47). Drawing on the words of Einstein and Infeld, he goes on to highlight the importance of problem-finding in addition to problem-solving in creativity (Csikszentmihalyi & Getzels, 1971, p. 47). Simon responds to Csikszentmihalyi's notion of problem-finding being a part of creativity by pointing out that while it is indeed a part of scientific discovery, it is not a creative process in itself. Problem-finding, according to Simon, is actually incorporated within problem-solving (Simon, 1988).

A further difference of opinion between Simon and Csikzentmihayli arose in relation to how creativity is viewed. While Simon explores creativity on an individual or personal level, Csikzentmihayli believes it has to be looked at in its historical and socio-cultural context. According to Csikszentmihalyi

(1988, 1996), creativity is made up of three main parts, the person, the field and the domain. It is only when society considers a person's contribution to be valuable that it is included into the domain. Creativity can thus, according to Csikszentmihalyi, occur only when all three systems are at work. In other words, Simon's view is that as long as the individual has found a solution that he believes to be satisfactorily creative, it can then be regarded as such. On the other hand, Csikszentmihalyi proposes that after the personal recognition stage, the individual still has the burden of obtaining approval and acceptance from the world (socio-cultural context) for his solution before it can be truly recognized as creative (Csikszentmihalyi, 1988).

Liu (2000) points out that creativity includes not only the problem-solving of an ill-defined problem, but also the problem-finding, heuristic-finding, solution-finding and recognition, while also encompassing motivation, originality and unconventionality. Csikszentmihalyi's and Liu's views on creativity thus bring into question Simon et al.'s notion of the creative potential of computers (Langley, Simon, Bradshaw & Zytkow, 1987). The computer is able to find not only a 'satisfactory' solution, but also the optimal solution that could fulfill all requirements for the personal level of creativity. However, on the socio-cultural level, as Liu points out, the computer's optimal solution may be regarded merely as a 'novelty', if recognition from society is not forthcoming (Liu, 2000, p. 268).

Akin (1986, 1990) notes that 'expertise' in problem-solving can be gained through the 'transformation of declarative knowledge into procedural knowledge' (Akin, 1990, p. 107). In other words, an individual becomes a more skilled problem-solver as he begins to use procedural knowledge more than declarative knowledge.

Based on the discussion above, (which does **not** claim to be an exhaustive study into the extensive research on creativity), one could argue that creativity does not include only the mysterious, unconscious talents possessed by a lucky few. Creativity can perhaps be seen as a conscious,

learned behaviour, with certain methods or process learned and carried out in creative problem-finding and problem-solving. As Lawson puts it, 'design is a highly complex and sophisticated skill. It is not a mystical ability given only to those with recondite powers but a skill which, for many, must be learnt and practiced' (Lawson, 2006, p. 14).

Because an aim of this system is to support creativity, the interfaces between the designer and the system must allow for the complexity described in this section.

3.1.1.1 Design

If you look up the word 'design' in any English dictionary, the first thing you will notice is that it is both a verb and a noun, which means it can refer to the final product resulting from the process. As a noun, design can include anything from a shoe to a spacecraft, suggesting that design as a verb must have an equal range of variability. The work of a fashion designer is clearly poles apart from the work of an aeronautics engineer. As Lawson points out, however, 'many forms of design deal with both precise and vague ideas, call for systematic and chaotic thinking, need both imaginative thought and mechanical calculation' (Lawson, 2006, p. 4). Lawson's notion of design falls in-line with Jones' conclusion that design is 'a hybrid activity which depends, for its successful execution, upon a proper blending of [art, science and mathematics] and is most unlikely to succeed if it is exclusively identified with any one' (Jones, 1992, p. 10).

As Simon puts it, 'everyone designs who devises courses of action aimed at changing existing situations into preferred ones' (1996, p. 111). However, as Galle pointed out, Simon's definition is far too wide, because while it does incorporate most forms of design, it could also include (to use his example) the plan to pick one's nose when no one is looking (Galle, 2008). Galle went on to suggest the problem with Simon's definition is that it focuses on planning and not on the actual production. Jones provides the shift of focus, when he defines design (as the verb) as 'to initiate change in man-made things' (1992, p. 4).

It soon becomes clear that, as Buchanan said, 'battles over the correct definition of design are fruitless' and 'the field will be vital as long as definitions come and go in the work of inquiry' (2004, p. 15). Lawson appears to hold a similar view to Buchanan's when he say that 'we shall never really find a single satisfactory definition but that the searching is probably much more important than the finding' (Lawson, 2006, p. 33). According to Buchanan, 'definitions serve the purpose of shaping a particular line of inquiry' (2004, p. 15). It is unsurprising then, that there are so many different definitions of design because 'design situations vary not just because the problems are dissimilar but also because designers habitually adopt different approaches' (Lawson, 2006, p. 12).

Design obviously means different things to different people. Rather than dismiss anyone's interpretations of what design is, and without attempting to define design anew, we come to the realisation that we can draw on elements of most definitions that, as Buchanan said, serve our purpose (2004). As Lawson states, 'To understand fully the nature of design it is necessary not only to seek out the similarities between different design situations, but also to recognize the very real differences' (2006, p. 29).

3.1.2 Design methods

Design methodology as an area of study is widely held to have grown out of the first 'Conference on Design methods', held in London in 1962. Cross (1999, p. 63) points out that this was not the actual birth of research into this area because 'the earliest reference in design methodology literature is probably Zwicky's work on morphological methods published in 1948'. However, the conference in 1962 was when design methods 'received substantial academic recognition' for the first time (Cross, 1993). Design methods, according to Cross (1993, 2001, 2007) had begun to emerge as a result of the attempts to apply 'novel, scientific, and computational methods to the novel and pressing problems of the Second World War' (Cross, 2001, p. 50). The design methods movement sought to make design more 'scientific', based on science, technology and rationalism (Cross, 1993, 2007). According to Cross, the desired design science was envisioned to be

a rational, wholly systematic and explicitly organised approach to design (2001).

Christopher Alexander and John Christopher Jones' findings could arguably be described as the greatest contributions to the first generation of the design methods movement (Janssen, 2004). Alexander's design method advanced the theory that the computer could play a large role in sorting and coordinating the broken down smaller components of a problem and could thus resolve the conflicts in each group (Broadbent, 1981). The design methods were viewed on the basis of a systematic, rational and scientific approach that aimed at excluding all forms of subjectivity. And yet, if we look at design as 'a hybrid activity … a blending of [art, science and mathematics] … [that is] most unlikely to succeed if it is exclusively identified with any one' (Jones, 1992, p. 10), then the assumption within that work is that any attempt to develop a wholly scientific method was bound to fail. As Gregory (1967) notes:

> … the scientific method is a pattern of behaviour employed in finding out the nature of what exists, whereas the design method is a pattern of behaviour employed in inventing things of value which do not yet exist. Science is analytic; design is constructive" (cited by Cross et al., 1981).

In a similar vein, Herbert Simon argues that while science deals with the way things are, design focuses on how things 'ought' to be (Simon, 1996). While in 1969 (the first edition of The Sciences of the Artificial), Simon calls for an 'intellectually tough, analytic' 'science of design', he tempers this when he writes that the design process should also be 'partly 'formalizable', partly empirical [and] teachable' (Simon, 1996, p. 113).

By the beginning of the 1970s, the concept of a design science was being rejected by many. As Harris (2009) writes, 'by the late 1960s, [Rittel] had discovered many reasons that [the first generation of design] methods were fundamentally problematic' (Harris, 2009, p. 96). Rittel and Webber point out that design and planning policy problems are 'wicked problems' that have no definitive formulation and thus cannot be solved absolutely

correctly. Science, on the other hand, deals with 'tame problems' that have definitive and objective answers (Rittel & Webber, 1973). It became clear that 'design science' was a contradiction in itself. It was not long before this design method was criticised for its mechanistic, inhuman approach and for denying the creative nature of the design by simplifying the problem (Janssen, 2004). Even the founders themselves were disillusioned by the movement. Alexander, who had conceived a rational architectural and planning design method 'disassociated' himself from the movement saying that there was little in design methods that had 'anything useful to say about how to design buildings', (quoted in Cross, 2007, p. 1). Jones also turned his back on design methods saying that he disliked the 'machine language, the behaviourism, the continual attempt to fix the whole of life into a logical framework' quoted in (Cross, 2007, p. 2).

With two such influential voices being so critical of design methods, the movement could very well have foundered. However, as Cross (2007) points, out the design method movement was saved by Horst Rittel's assertion that the design methods that had emerged in the 1960s were only the first generation and that a second generation had already started (Rittel & Webber, 1973). Rittel was, in fact, already developing the second generation of design methods. One of Rittel's main focuses in developing the new methods was on transparency (Harris, 2009). He was adamant that the second generation of design methods 'should lead to a situation where every step of the planning process is understandable and communicable or 'transparent' (Rittel, 1972, p. 394).

The second generation of the design method movement that emerged recognised that the notion of an 'optimal design solution' was inconsistent with the ill-defined 'wicked problems' of design. For optimisation, the problem would have to be over-simplified (Simon, 1996). As a result, there was a move toward design decisions that are 'good enough' or 'satisfice', a concept introduced by Simon, which acknowledged that a solution would 'rarely be optimal in the real world, but … will often be satisfactory' (Simon, 1996, p. 27). Another development was in the way that the designer's role

was viewed, especially in the fields of architecture and planning. As Lawson points out, the design problem does not always come from the client as one might at first assume, but anyone who might be affected by the design, including the client, the users, the designer himself and the legislators (Lawson, 2006). Because the design problem can be influenced by all the parties affected, it suggests that the design solution would also be. Thus, the second generation of design methods included the realisation that designers should be viewed more as 'partners with the problem "owners"' (clients, customers, users, the community)' and the process of design was thus seen to be more 'argumentative' and 'participatory' (Cross, 2007, p. 2). Another major realisation made by the second generation of the design methods movement was that designers have preconceptions that they cannot avoid bringing to the design problem. As a result, they are unable to solve the design problem completely objectively (Cross et al., 1981).

Although the second generation of the design method movement differed from the earlier methods, it still 'framed design as a problem-solving endeavour and remained in the broader design science framework' (Janssen, 2004, p. 38) However, as Janssen points out, many of the second generation methods were only slightly different from the methods of the first generation (Janssen, 2004). Alexander, for instance, realised that his previous assertion that the design process should be objective and strictly scientific was flawed. He subsequently modified his method to exclude the computer (Alexander, 1971). The 'Pattern Language' method was introduced (Alexander, Ishikawa & Silverstein, 1977). It consisted of patterns that expressed the relationship between a context, a problem and a solution. The designer who, by this method, could be the client, user or the community rather than just the architect, would build up a design solution by applying the pattern in a structured way. Thus, subjectivity was re-recognised, only this time it was the designer's in addition to the user's input that was sought.

During the 1980s, as many architects became interested in symbolic forms and functionalism, the rationalistic approach to design was almost

completely rejected (Janssen, 2004). Cross et al. argued that 'design simply is not like science' and proposed instead that 'design should be viewed as a technological activity' (Cross et al., 1981, p. 200). As Cross writes, 'technology involves a synthesis of knowledge and skills from both the sciences and the humanities, in the pursuit of practical tasks; it is not simply 'applied science', but 'the application of scientific and organized knowledge to practical tasks' (Cross, 1982, p. 222).

Today design methods, whether rational approaches or random and 'accidental' techniques, are developed and understood as a broader category of methods designed to generate solutions for the ill-defined problem of a given design task. Some theorists argue that design actually deserves a space of its own, while problem-solving should be seen as a different and separate discipline (Glanville, 1998). Generally speaking however, all approaches to design recognise that the human element of creativity plays a part in the design process. As discussed in the previous section, creativity, whether in human thinking behaviour, in art, architecture or science still remains mysterious (Liu, 2000). Liu described creativity in the design process as a 'huge ill-defined problem' which can be broken down into 'problem-finding, heuristic-finding and solution-finding', as opposed to the traditional view of the design process as being 'problem solving' (Liu, 2000, p. 267). Solutions to the design problem are not easily found however, because there is often an insufficiency of information regarding the requirements that the design should fulfil. This often means that the designer will have to discover, create and invent the required information. This appears to show that the 'solution-finding' approach to the design process actually requires a more creative approach, rather than a scientific one.

3.1.3 The role of the designer

'Designing is something that all people do; something that distinguishes us from other animals, and (so far) from machines' (Cross, 1999, p. 25). Lawson quotes Karl Marx in his book *Das Kapital* from 1867, where he made the distinction between human 'design' and the building activities of

a bee, is that the human first 'imagines' what he is going to make, thus, 'at the end of every labour process we get a result that already existed in the imagination'. Because Jones defined design as 'the initiation of change in man-made things', the earliest designers can be seen to be the craftsmen who were skilled in making things (Jones, 1992). As Cross points out, humans have a long history of design, as attested by the achievements of previous civilisations (Cross, 1999). Lawson agrees with this and points out that many incredibly complex and elaborate artefacts have been designed and made throughout history with no theoretical background in design (Lawson, 2006).

3.1.3.1 Design through production

'Craft' or 'blacksmith designs' were produced by craftsmen who 'designed as they made', 'working to undrawn traditional patterns handed down from generation to generation' (Lawson, 2006, p. 21). As Jones says 'hidden in the apparent simplicity of primitive craftwork, [is] a subtle and reliable information-transmission system that is probably more efficient than design-by-drawing' (Jones, 1992, p. 15) Both Jones and Lawson cite George Sturt's book *The Wheelwright's Shop (1929),* in which Sturt provides an account of nineteenth century wagon-making, as an invaluable source for the exploration of the role of the designer (Jones, 1992) (Lawson, 2006). The 'traditional', 'craftsman' designer did not draw any plans prior to designing, he designed instead by combining his 'know-how', the knowledge that he had gained from his own experience with the 'know-that' that was transmitted to him, not through formal education, but through an apprenticeship, during which he would have learnt how to make the product by actually making it. The 'know-that' knowledge 'learnt' from a skilled practitioner was not based on formal theoretical principles, but was actually the 'know-how' of previous generations passed on as an accepted fact. The knowledge transmitted from previous generations would thus be the result of many generations of trial-and-error (Jones, 1992). Most often, a craftsman would work with a 'curious combination of constructional skill and theoretical ignorance' (Lawson, 2006, p. 21). As

Jones points out, the product of craft design itself contains all the required knowledge that has been learnt through the generations. The product, and thus the store of knowledge, is changed only when an error is recognised and subsequently corrected or when new demands dictate a change in 'design' (Jones, 1992). There is an obvious weakness inherent in this design process — the designer's 'know-that' is based on the designed form as a whole. This raises two problems; first, if one aspect, either functional or aesthetic, needed to be changed, the designer would not know whether changing that one thing would compromise the entire design. The second problem is that any change in the design has to be made with only one thing at a time being changed. Any change or development was thus a hit-and-miss process of trial-and-error.

3.1.3.2 Design through drafting

The Industrial Revolution of the late eighteenth and nineteenth centuries brought with it the 'professionalisation of design' (Lawson, 2006, p. 23). Christopher Alexander pointed out that the 'unselfconscious' craftsman designer was bound to change into the 'self-conscious' professional at a time when society was undergoing a sudden and rapid change (Alexander, 1964). As Lawson points out, the craftsman's evolutionary process could not keep pace with the very rapid and great developments in materials and technologies that were occurring during this period (Lawson, 2006). Thus, the role of the designer began to be separated from the 'maker' by the use of 'scale drawings in place of the product as the medium for experiment and change' (Jones, 1992, p. 20).

As Lawson points out, design-by-drawing allows the designer greater room for creativity. His drawings became 'a part of the very thinking process … which we call design' (Lawson, 2006, p. 26). In addition, the new process allowed the designer to change more than just one part of his design at a time (Jones, 1992). The designer's drawings were also used for production and made possible the division of labour that allowed for the plans to be drawn for the much bigger things that a single craftsman could not have made on his own. Having more than one craftsman work on a production

also meant that the speed at which things could be produced was greatly increased (Jones, 1992). The designer, who was now separated from the production process, was conscious of the change in his role and, as Alexander wrote, 'recognition of his individuality has a deep effect on the process of form-making' (Alexander, 1964, p. 59).

Design-by-drawing had weaknesses, which the design methods movement of the 1960s called into question. Although the design drawings could show the designer and the client what the final product might look like through relatively-accurate models, it could not show how the product would perform (Lawson, 2006). Lawson offered, as an example of this 'poor representation' of the functionality of a design, the new-found ability (in the 1960s) to build high-rise blocks. Although the designers' drawings could show an accurate depiction of the visual aspects of the blocks, they failed to show the 'social problems which were to appear so obvious years later' (Lawson, 2006, p. 27). As Jones points out, the designer's drawing lacked the ability to communicate the needs and problems of both the users and the manufacturers (Jones, 1992).

3.1.3.3 Design through science

By the 1960s, the shortcomings of the design-by-drawing method became increasingly apparent, particularly in architecture (Lawson, 2006). More scientific methods of modelling the final design were sought. It was believed that designers should be more like scientists, who were believed to have precise methods that led to their accomplishments (Cross et al., 1981). Scientists' work could, after all, 'be replicated and criticised and their methods were above suspicion' (Lawson, 2006, p. 28) The first generation of the design methods movement envisioned a 'design science' that would be more rational, wholly systematic and explicitly organised (Cross, 2001). The designer would thus be required to take on a more rational, systematic and explicitly-organised role within that design process. The notion of a 'design science' aimed to exclude the subjectivity of the designer and threatened to diminish, if not completely eliminate, the individuality and identity of the designer.

As Cross points out, even during the 1960s, when a more scientific approach to design was being called for, it was obvious that designers could not just take on the methods of the scientists (Cross et al., 1981). The ill-defined, 'wicked problems' facing designers could not be solved 'scientifically' (Rittel & Webber, 1973). Designers and scientists not only face vastly different problems, but also have widely divergent interests and goals. Science seeks to find out the nature of what exists, while design seeks to invent things of value. 'Science is analytic; design is constructive' (Cross et al., 1981, p. 195) quoting Gregory. Even just the notion of the designer's role as a practitioner of science was short lived.

By the 1970s, Hillier et al. suggested that the first generation of design methodologists had failed to take into consideration a change in the epistemology of science itself (Cross et al., 1981; Hillier, Musgrove & O'Sullivan, 1972). They claim that there had, in fact, been an acknowledgement of the fact that the preconceptions or 'pre-structures' were an inevitable part of the scientific methods and were therefore also a part of design methodology. Designers and scientists alike bring their pre-conceptions to any problem they attempt to solve. Hillier et al. argue that the pre-structure had been recognised as a part of the scientific methods; pre-structure in design could not be seen as 'unscientific' in design methodology (Cross et al., 1981; Hillier et al., 1972).

Broadbent modified the 1960s design process by including 'preconceptions' in the synthesis stage of the process (Broadbent, 1988). He argues that preconceptions explain why different designers given the same brief and the same analysis, reach different solutions according to their preconceived stance. These preconceptions play an essential role in the creative aspects of the design process (Janssen, 2004). Three types of preconceptions, discussed in Section (4.5), can be identified.

- Guiding Principles

  Variously called the designer's 'paradigmatic stance' (Broadbent, 1988), 'guiding principles' (Lawson, 2006) and 'theoretical position'

(Rowe, 1998), the guiding principles reflect the philosophical beliefs, cultural values and background of the designer. This stance evolved and developed all the way through the designer's practicing career (Janssen, 2004). The guiding principles part of the designer's preconceptions influence the capture of the design concept.

- Primary Generator

  Various scholars attribute the formation of these initial ideas to a variety of sources, the 'primary generators' (Drake, 1979), 'enabling prejudices' (Rowe, 1998), 'concept or parti' (Lawson, 2006) and 'generative concept' (Frazer, 1974) (Frazer & Connor, 1979) (Frazer, 1995). The primary generator influences the conceiving of the design schema.

- Design Schema

  The schema, as Lawson (2006) sees it, is the accumulated experiences of an individual, which he/she then uses as a reference for understanding and interpreting current or future events. Frazer, on the other hand, views the schema in a more design-specific context and describes the schema as a 'highly personalised but generic methodology' used by designers (2002, p. 259). Seen from this point of view, the schema is a design model that can be adapted to different design situations and problems and can be used by the designers as a part of their design process. According to Frazer (2002) and Janssen et al. (2002), the schema thus characterises the designer's personal style as an abstract conception of the common features shared by all their designs. As Janssen points out, the schema is developed throughout the practising career of a designer, as a direct reaction to the 'niche' environments that are 'a range of design environments that are similar to one another' (Janssen, 2004, p. 163).

What becomes evident is that every design with its related design process is by its nature a unique undertaking. Not only are the aims of each design

process unique but so too are the design environment and design team who have constantly developing preconceptions. The challenge posed by the dynamic nature of design mean that any attempt to develop a design method or design system should allow for the 'uniqueness' of each design project.

3.2 Computer Aided Architectural Design (CAAD)

As discussed earlier in Section (3.1.3), the emphasis placed on the role of the designer within the design process, has undergone a number of changes, from the 'design through production' involving 'craft' or 'blacksmith designs', to the 'design through drafting' witnessed in the post-industrial revolution of the late eighteenth and nineteenth century, which saw the 'professionalisation' of design, to the 'design through science' of the 1960s, which aimed at excluding the subjectivity of the designer. 'It is apparent that design is strongly dependent on the tools used and, reversely, tools have a profound effect in design' (Terzidis, 2006, p. 25). The designer's role today is closely related to the role of the computer in the design framework (Janssen et al., 2002). As the attitude towards computers alters, so does the attitude toward computer applications in design. This, in turn, leads to altered design methodologies that incorporate the use of the computer and so, ultimately, the role of the designer is also altered. Yehuda Kalay (2006) points out that 'almost every time society invented new tools, methods, and techniques to manufacture and distribute the products needed for its survival and growth, these inventions impacted society itself economically, culturally, politically, and in many other ways' (Kalay, 2006, p. 357). He argues that the technological revolution is impacting the profession and discipline of architectural design by 'transforming the current strictly-hierarchical design process into a network of design, manufacturing, marketing, and management organisations, where the responsibility for design operations is distributed across multiple professions, organisations and geographic locations' (Kalay, 2006, p. 358).

Although Terzidis says that the original role of computers in architecture was to 'replicate human endeavours and to take the place of humans in the design process' (2006, p. 60), the first reaction to the introduction of computers automating the production part of the design process, was that the 'computer is just a tool'. Later, the analysis and simulation intelligence of the computer shifted this attitude, and computers were viewed as a 'smart tool' that would be 'intelligent assistants to designers, relieving them from the need to perform the more trivial tasks and augmenting their decision-making capabilities' (Terzidis, 2006, p. 60). Today, the greatly advanced capabilities of computers and related software are changing the dismissive and negative attitudes to the computer. Computers in architectural design are used for drafting and modelling at one end of the spectrum to the form-based processing of architectural information at the other end of the spectrum. There is every indication that this 'tool' will become one of the main components at the centre of the design process (Janssen et al., 2002). The computer will not, however, replace the human designer. As Terzidis (2006) points out, the computer is not a human mind. The human mind cannot be replaced by the computer, because a computer can only make a limited number of decisions based on set data that has been entered by the human operator, thus the computer is not and never will be a designer. The aim should not be for the computer to become the designer. As Kalay (2004) notes, while architectural researchers see computer-aided synthesis as the key to rational design, the practitioners in the field are suspicious of the use of computers in architecture (for actual computer-aided synthesis as opposed to visualisation alone) because they see this as potentially removing the human essence from architecture. Surely the computer in architectural design should complement rather than threaten the role of the designer. It should not seek to replace the designer, because the human cognitive process must still be acknowledged as the main driving force behind the design solution. The computer should be seen as a 'counterpart to human imagination, a source of ideas and a portal into another world new to the human mind' (Terzidis, 2006, p. 41).

Although a great deal of architectural practice has changed from the traditional, manual tool-based process to a design process driven by computers, it has not reached its full potential. Terzidis (2006) argues that, to a certain extent, this is due to a gap in the computational education of the architects that is compounded by the fact that most of the literature on digital design is confusing and there are very few examples of computers being used to their full potential as design tools. As Terzidis puts it, 'most of the researchers in CAD were primarily concerned with the technicalities of converting design ideas into digital tools, none, if any, was also concerned with using those tools to actually design' (Terzidis, 2006, p. 54).

3.2.1 Computation vs. Computerisation

If no distinction is made between 'computerisation' and 'computation' and the term 'computing' is taken to include both, one can say, as Kalay (2004) does, that computing has been a part of architectural design since ancient times. Kalay bases this view on the fact that there is evidence that the calculation of sizes, proportions, areas and volumes using geometry had been used in architectural design and construction as long ago as 5000–2000 BC in Mesopotamia, 2750–1500 BC at Stonehenge and 2570–2500 BC in Ancient Egypt and, with the use of geometry, it gained in importance during the Renaissance when proportion became a focal point in all architectural design (Kalay, 2004). Taking this view of 'computing', that covers the actual deduction of solutions to a problem, computing, through computation, in architectural design can be traced throughout the ages. It is clear that there is a need for, and the presence of computation, in all architectural design from ancient to present times.

However, a distinction can and should be made between computation and computerisation in architectural design. Terzidis (2006) does distinguish between the two terms by highlighting that computation is focused on the exploration of ill-defined and indeterminate processes and attempts to copy and extend the mental processes of the human brain. Computation is, in fact, the process of calculating, determining or working-out a solution through mathematical or logical methods, which can include

'rationalisation, reasoning, logic, algorithm, deduction, induction, extrapolation, exploration and estimation' (Terzidis, 2006, pp. xi, 57). Computerisation, on the other hand, deals with the 'automation, mechanisation, digitisation and conversion' through the entering, processing or storing of information into a computer system (Terzidis, 2006, pp. xi, 57). Confusion between computation and computerisation occurs, because most computation, that can of course be done manually, is carried out using a computer. Ironically, as Terzidis (2006) points out, computers in architectural design are most often used for computerisation even though their most valuable contribution is in computation. The computational capabilities of the computer can greatly aid human designers who cannot compute extreme quantitative complexity, and yet the computer needs the human designer as 'computation processes fail to justify consciously even simple decisions' (Terzidis, 2006, p. 29).

As the architectural design problem is often vague, indefinite and uncertain, the actual process of form-making results from a 'combination of thoughts that lead to the inception of a form' (Terzidis, 2006, p. 40). This is precisely where the importance and potential of the computational abilities and the algorithmic logic of the computer become clear. The vague, indefinite and uncertain design problem can be translated using algorithms as algorithms use a language with a vocabulary, syntax and meaning that, unlike a human language, is 'dependent on the features of a digital computer' as opposed to being dependent on the 'communicative power between a human and a computer' (Terzidis, 2006, p. 40).

The use of computers in the domain of architecture and the building and construction industry was first introduced mainly to aid in the analysis of information for engineering. As Kalay (2004) points out, however, only certain inputs into the design process were processed by a computer, while the overall design process followed the traditional manual route. Although the computation of even some of the processes aided in solving complex calculations, the manual entering of data into the traditional design process

very often led to mistakes being made and, ultimately, had the potential to hinder rather than aid the design process (Kalay, 2004).

By 1963, Sutherland had introduced the use of the computer as an interactive graphical design tool with his Sketchpad program (Kalay, 2004; Kotnik, 2010). Kalay describes this as the invention of 'both the modern concept of computer-aided design and the tools to implement it' (Kalay, 2004, p. 65). Sutherland's Sketchpad proved that the computer could be used not only for computation, but also for drafting and modelling. In their article *Geometric Modelling: A survey*, Baer, Eastman and Henrion, (1979) discuss the 'recent' (that is, recent to the late 1970s and thus over 30 years ago) recognition that computers can be used in much the same way as drawings for representation. They highlight that this use of the computer for the representation of forms was not limited to merely displaying a drawing, which would involve the inclusion of the drawing's inherent limitations, but that the computer could be used to actually represent the 3D form itself (Baer et al., 1979). Using the computer for modelling also means that the computer can represent the same information in a single model that would, if using a traditional manual design process, involve the production of many drawings, specifications and engineering data. 'Such a model coupled with other data can allow automatic programs to prepare data for engineering and, if necessary, automatic drafting' (Baer et al., 1979, p. 253).

Three decades later we see that the use of the computer, if not for the huge potential they offer in computation but for graphical representations at least, has become vital for architectural design. The fact that computers in architectural design are used mainly in the automation of traditional design processes such as drafting and modelling (Kotnik, 2010) means that most of their potential to enhance and add value to architectural practice is being wasted. Kalay (2004, p. xvi) goes so far as to say that the use of computers in architectural design has:

> … had — so far — relatively little qualitative impact on the profession of architecture itself and on the way buildings are constructed and used when in fact [they have] the potential to reinvent the architectural design process itself.

3.2.2 Digital architectural design methods

Much research has been carried out into the development of computer-aided architectural design, or 'digital design', which, according to Oxman, had by 2003 come to represent the 'concept of non-standard, non-normative, non-repetitive design' (Oxman, 2006, p. 232). As Kotnik (2010, citing; Oxman, 2006) points out, the notion of digital design still lacks a clearly-formulated theoretical foundation and its underlying basis is greatly influenced by ideological positions. A common and easy mistake to make is to see the mere use of computers in architectural design as 'digital design'. The use of computer applications is not, in itself, 'digital design'. As Terzidis points out, 'digital, in the true sense of the meaning, is about the reduction of a process into discrete patterns and the articulation of these patterns into new entities to be used by a computer' (2006, p. 39).

3.2.2.1 Kalay's methods

Yehuda Kalay, in his book Architecture's New Media (2004) explores the principles, theories and methods that were first developed and used by human designers and which have subsequently been transferred to the world of computer-aided design. According to Kalay, computation in design is patterned on traditional, habitual methods, and the changes made to these methods to take advantage of the processing and storage capacity of computers has not fundamentally changed them. The difference being that the computer has processing and storage capabilities that allow the designer to generate and test a much larger number of potential solutions and, at the same time, also allow for databases of design rules and precedents to be stored and searched. Kalay (2004) looks at the use of computers in design through three methods, procedural, heuristic and evolutionary.

Procedural methods, according to Kalay, were the 'first to be employed in the quest for computational design synthesis' (Kalay, 2004, p. 235). Procedural methods allow the human designer to set specific desired local conditions, for example the relationship between the variables. The computer can then analyse and test these specified conditions over a much larger set of variables. If applied 'judiciously', complex forms can be generated from a small number of specified conditions. Using procedural methods, however, still requires human observation to detect, exploit and avoid global opportunities and pitfalls (Kalay, 2004).

Kalay divides the use of the computer in procedural methods into three sub-methods, complete enumeration, space allocation and constraint satisfaction. Complete enumeration is the design method based on the notion that the more potential solutions that can be chosen from, the better (that is, if it was possible to chose from all possible solutions, this would be better than choosing from only a few). The computer can greatly aid in attaining complete enumeration by using a systematic method to find all the geometric pattern combinations possible for a given problem. The designer can then choose the one that is the best for the design context and function. Complete enumeration is only useful if the number of potential solutions is too great for a human to remember and not too many to for the computer to process. The vast number of combinations ('combinatoric explosion' as Kalay (2004) calls it) possible for generating floor plans and that the complete enumeration method could not be effectively used in this context led to what Kalay calls 'space allocation' in his procedural methods.

Also known as 'automated floor plan generation', 'automated spatial synthesis' and 'quadratic assignment formulation', space allocation is, according to Kalay, 'one of the most interesting, difficult, and controversial areas of computer-aided architectural design' (2004, p. 241). Space allocation seeks to layout the spaces in a building according to rational principles. Given a set of spaces and the desired adjacencies between them, which can be weighted, space allocation seeks to find a layout with the minimum distance between the spaces that have adjacency requirements.

Although it is one of the first design problems to be computationally synthesised, space allocation 'still only works in very limited areas of architectural design, primarily ones in which quantifiable design objectives can be formulated and where the nature of the solutions is relatively well understood' (Kalay, 2004, p. 241). Space allocation methods are not able to produce actual floor plans and are not optimal solutions for the design problem, because they consider only the distance criterion and ignore all other design criteria (Kalay, 2004). When the distance criterion is not the main focus and other criteria, such as lighting and the site conditions are as important, many more constraints are added to the design problem.

The addition of constraints to the design problem introduces what Kalay (2004) calls the 'constraint satisfaction' procedural method, which can be approached by either 'linear programming' or 'successive approximation'. Using linear programming, constraint satisfaction can be attempted by using operations research, which is a mathematical (numerical or graphical) way of 'solving a system of linear inequalities, each representing one constraint on the solution' (Kalay, 2004, p. 248). The problem that arises is that criteria are very often conflicting. To handle this problem, conflicting constraints need to be 'managed … [so that] not all constraints need to be solved at once' (Kalay, 2004, p. 249). Rather, the constraints can be arranged in a dependency hierarchy so that each constraint is handled one at a time and if the solution at an earlier level produces an unworkable situation in a subsequent level the first solution will be modified. The 'successive approximation' method can also be used to deal with multiple conflicting constraints. The constraints are not all satisfied at the same time, neither are they dealt with one after another, instead a solution is developed that will satisfy most of the constraints in a way that is 'satisficing' (the term coined by Herbert Simon (1996) to describe solutions that are not optimal but are sufficiently satisfactory, discussed in Section 3.1.2) by beginning with a solution that can be gradually improved to satisfy as many constraints as possible.

The next design method in Kalay's categorisation of design methods that incorporates the use of the computer is heuristic methods. Design problems are not tackled with a full understanding of the task at hand, which is what the procedural methods require (Kalay, 2004). As discussed in Section (4.5), the designer brings his preconceptions to each design problem. These preconceptions are the accumulation of the designer's personal and professional experience and expertise and thus many methods of design are, in essence, heuristic. While procedural methods are more exact, heuristic methods cannot guarantee that a solution, let alone an optimal solution can be found. As Kalay points out, although heuristic methods have 'logical gaps and inconsistencies' because they rely on inexact reasoning and knowledge, they can solve problems that could not be solved by procedural methods, because they deal with 'a global, holistic view of the problem … rather than the localised one most procedural methods rely upon' (Kalay, 2004, p. 256). Kalay divides the heuristic methods into four, analogical, case-based, expert systems and shape grammars.

The analogical heuristic methods are based on the idea that design solution can be synthesised by learning and borrowing from other knowledge fields when there seems to be a shared relevance. In the case of traditional design methods, the designer could observe and borrow from other fields. However, with computationally-based heuristic methods, the 'source of the borrowed knowledge and its method of adaptation must be formalised in some manner' (Kalay, 2004, p. 256). Heuristic methods in design have borrowed from the fields of electrical engineering and mechanical engineering. For instance, an architectural floor plan can be represented in a similar way to an electric circuit using a network of nodes and edges to represent architectural elements and their respective relationships. While this kind of representation would not show the geometry of the floor plan, the doors, windows and so on, it can be used for analysis such as adjacency requirements. Borrowing from mechanical engineering, physically-based modelling uses the 'principles of dynamic motion and geometric

deformation to rigid and non-rigid objects for the purpose of simulating realistic behaviours and visual effects' (Kalay, 2004, p. 258).

Case-based heuristic methods, according to Kalay's categorisation, take a previously-found solution to a similar design problem and modify it with the requirements and conditions of the problem at hand. The case-based method, supported by the fact that most design problems arise in an attempt to answer the needs of humans that are very often similar, assumes that the problem being faced is not fundamentally different from the problem faced in the past. The case-based method is also based on the assumption that modifying a previous solution is better than solving the current problem from the beginning. Kalay points out that this assumption is supported by the 'added information embedded in the 'whole' compared to its 'parts"' (Kalay, 2004, p. 261). In other words, a designer can gain from using a previously-produced solution because it encompasses a solution to the 'whole' and although he may have the knowledge needed to solve parts of the problem, this knowledge may not be enough to solve the over-all problem in the most optimal way. Using the case-based method is clearly limited by the availability of prior design solutions and the architect's knowledge and memory of solutions that do, in fact, exist. Computer databases (aided by keywords, solution attributes and so on) can greatly aid in retrieving prior solutions and can be used to a certain extent to adapt them. Building up meaningful case libraries does, however, pose a very big problem. The 'Software Environment to support Early phases in building Design' (SEED) is a system used to collect cases throughout the design process. The designers can choose any solution that they generate as a 'case' that they can store in the system (Kalay, 2004).

The third heuristic method outlined by Kalay is the expert systems, which use the knowledge and experience of other designers in the form of design rules rather than in the form of prior design solutions. Expert systems are 'computational constructs designed to capture and represent the knowledge of an expert in the form of heuristic 'rules of thumb' — encapsulated 'chunks' of professional practices, common sense, shortcuts,

insights and other special-case reasoning characteristic of highly-experienced professionals' (Kalay, 2004, p. 268). The generalised rules and inference used in expert systems, as opposed to the hard-coding of factual data used in procedural methods, actually means that expert systems are more flexible than procedural programs, especially when dealing with situations where there is insufficient information or uncertainty (Kalay, 2004). As Kalay argues, expert systems 'can even resolve problems arising from conflicting data, by incorporating personal (or expert-specific) preferences' (Kalay, 2004, p. 268). There is, of course, a down side to this in that expert systems are not as reliable as procedural methods; nevertheless, if a solution does exist, expert systems are expected to find it faster than a procedural program 'as their heuristic reasoning can exploit shortcuts to reach conclusions rather than have to test all solutions exhaustively' (Kalay, 2004, p. 268).

The final heuristic method detailed by Kalay (2004) is the shape grammar method. This method uses a 'system of rule-based geometrical constructions for designing shapes based on strict compositional rules' (Kalay, 2004, p. 272). A grammar is a set of rules that regulate a process or knowledge. The rules of shape grammars can be used on a geometrical construct to modify its constituent shapes through geometrical transformations, such as addition, subtraction, translation, rotation and mirroring. Shape grammar rules are geometrical constructs (points, lines, planes or volumes) and can be taken from a particular school or corpus of architecture. This allows the 'essence' of the work to be extracted and used according to the requirements of a design problem. This is especially useful if 'the body of work is to be extended' in cases of restoration or preservation, for example (Kalay, 2004, p. 274). Shape grammars have a generative advantage over other shape synthesis methods, because the designer can decide to use any rule throughout the design process. The shapes are not predefined but are actually formed through the application of the grammars. As Kalay points out, shape grammars, like any other

mechanical generative model, rely on the recognition of shapes, rather than on the computer's ability to actually generate the shapes.

The evolutionary methods, the last in Kalay's (2004) categorisation of computational design synthesis, came about as a result of the birth of artificial intelligence (AI). With the emergence of artificial intelligence, new methods and tools for computational architectural design synthesis became possible. The evolutionary methods attempt to solve design problems by using search algorithms that can search large unstructured problem spaces and produce solutions that are 'creative', 'novel' and 'surprising'. At the heart of the evolutionary methods are genetic algorithms (GAs). Computer-simulated evolutionary processes are based on the same principles of 'survival of the fittest,' on which natural Darwinian evolution is based. As Kalay (2004, p. 283) explains it,

> [Gas] consist of a population of solutions, called phenotypes, which are represented by their genetic code, or genotypes … a fitness function is used to select the fittest phenotype from within the population of candidate solutions ...[and] their corresponding genotypes are used to create new, and conceivably better, populations of solutions, through mating and mutation.

Kalay points out that the only major drawback in natural evolution 'is the length of time it takes to complete each cycle ('generation') and evolve a fitting solution' (Kalay, 2004, p. 282). On the other hand, with computer-simulated evolutionary processes, the speed of the digital systems means that the 'evolutionary' process can be speeded up and thousands (sometimes millions) of generations can be processed in a very short time.

'Artificial Neural Networks' (ANNs) are another evolutionary method. ANNs developed from the cognitive theory of connectionism, 'which argues for processing information as patterns, by means of networking many independent small signal processors, rather than by executing explicit instructions one at a time' (Kalay, 2004, p. 287). ANNs have pattern

recognition abilities, which makes allows them to recognise innovative design solutions (Kalay, 2004).

3.2.2.2 Oxman's Models

Oxman, in her article, *Theory and design in the first digital age,* suggests that one of the main questions that should be resolved is whether digital design is, in fact, the re-invention of the architectural design process that Kalay (2004) argued was possible with the aid of a computer. Whether or not digital design is indeed a unique phenomenon, Kotnik (2006) points out that the computer is the 'principal design tool' for digital design. Perusing the notion that digital design may actually be a unique form of design, and using conceptual categories of design research methodology, Oxman (2006) formulated a framework or a series of models to aid the theoretical approach of digital design. As Kotnik (2010) describes them, Oxman's models are 'paradigmatic classes' of digital design models, grouped together by 'various relationships between the designer, the conceptual content, the design process applied, and the design object itself' (Kotnik, 2010, p. 12). Oxman's digital design models, which 'demonstrate a successively structured development' are classified as CAD models, formation models, generative models, performance models and integrated compound models (Oxman, 2006, p. 246).

Oxman's CAD models are based on the use of the computer rather than the traditional paper-based design processes. The designer using CAD 'deals with the geometric structure of *a priori* design objects' (Oxman, 2006, p. 254). The CAD models are sub-divided into three sub-models, descriptive CAD, 'generative-evaluation' (predictive) CAD and CAD descriptive (physical, virtual). The descriptive CAD model includes the earliest uses of CAAD, and is mainly concerned with the graphical representation of objects. The 'predictive' CAD model incorporates developments in CAAD and is concerned with the analytical processes for evaluation. The generative and optimisation processes of this model are neither formulated nor automated and are not linked directly to the representational and

evaluation modules. The 'physical-virtual' CAD model captures physical objects and translates them into a digital format and vice versa.

The digital formation model in Oxman's digital design framework is a complete departure from paper-based design and is a move towards the dynamic concepts of formation, as opposed to the more static concept of form. The digital formation models 'provide enabling design media for geometrical and topological control of variant formal generation within conditions of topological control' (Oxman, 2006, p. 254). The digital formation model is sub-divided into the topological formation model that centres on the use of topology and non-Euclidean geometry when designing. The associative design formation model focuses on parametric design and generative components. The motion-based formation model that has as its base the concept of dynamic design and focuses on animation, morphing and other motion and time-based modelling techniques.

Oxman's generative model 'is the design of, and interaction with, complex mechanisms that deal with the emergence of forms deriving from generative rules, relations and principles' (Oxman, 2006, p. 254). The generative design model is sub-divided into the grammatical transformative design models that use shape grammars to drive shape generation processes through transformation rules; and the evolutionary design model focuses on the development of design through the natural evolutionary process. According to Oxman, performance-based design models use 'digital technologies that support the generation of form resulting from design performance' (Oxman, 2006, p. 257). The performance model is sub-divided into performance-based formation models and performance-based generation models. Both performance-based models use performance simulation, but the performance-based formation models centre around the digital formation models described above, while the performance-based generative models focus on the generative design models. Finally, the integrated compound models incorporate what Oxman believes to be the 'future paradigmatic digital

design media that have potential implication for future design media' (Oxman, 2006, p. 260).

3.2.2.3 Kotnik's Frameworks

Kotnik (2006, 2010) formulated a framework of digital architectural design similar to that designed by Oxman (2006) discussed above. As Kotnik says, 'both studies show, that the digital way of designing and exploring architecture has to be seen as an extended form of expression that is interwoven with the non-digital in manifold ways' (Kotnik, 2006, p. 9). This would suggest that, far from the use of the computer in architectural design limiting the designer's creative expression, digital design could actually be seen, in many ways, as an extension of the creative expression of the designer. Oxman, to understand the 'uniqueness' of digital design and to create 'a medium to represent the syntax and content of models encountered in the analysis of digital designs', defines the unique properties and characteristics of digital design (Oxman, 2006, p. 239). While her approach centres on the 'nature of interactivity and type of control design processes', Kotnik's work is based on the Turing-model for computers (Kotnik, 2010, p. 39). His model is the 'conscious perception of the abstract Turing-model for computers' (Kotnik, 2006, p. 39). Kotnik's Turing-based model divides digital design into the representative, the operative, the parametric and the algorithmic levels and, as he points out, is 'an attempt to separate the morphogenetic process in architecture from other computational methods' (Kotnik, 2010, p. 43).

Kotnik's (2006, 2010) categorisation of digital design models is based on the designer's (operator's) level of awareness of the computational processes taking place. As Kotnik (2010, p. 7) describes it, '

> The degree of awareness of the computational background and its intentional use that can be seen as the defining characteristics of digital design and at the same time as a possibility to mark the threshold between digital and non-digital design.

The first level of digital design in Kotnik's Turing-based model, the representative level, is based on the concept of using the computer to describe the unchanging external state of form. At this level, Kotnik (2006) sees digital design as merely representative, because the computer, as an essential design tool, is not even considered. Kotnik sees this level of digital design as a waste of the potential offered by computers, because even when using the computer just as a representational tool for drafting or modelling, an algorithm is actually running mathematical descriptions of the forms in the background and yet the designer is unaware of them and is not able to make full use of their potential. Kotnik's representative level of digital design corresponds to Terzidis' description of the use of computers in design as 'marketing tools of strange forms whose origin, process, or rationale of generation is entirely unknown' (Terzidis, 2006, p. 40).

The type of geometric operations used by the computer for modelling distinguishes the operative from the representative level in Kotnik's Turing-based model of digital design. At the operative level, the computer carries out modelling operations in 'a pre-defined geometric way' (Kotnik, 2006, p. 31). The geometric operations being processed by the computer are 'explored in an architectural context in order to deform the classical formal language of architecture by means of controlled transformations' (Kotnik, 2006, p. 31). Unlike computer-aided design at the representational level, which is, in effect, 'architectural design crafted in the traditional way' with the computer being used merely for representation and visualisation, the operations being carried out at the operational level 'were not used widely in architecture because of the inherent increase in geometric complexity which limits the ability to handle them in an efficient way by means of drawing as well as imagination' (Kotnik, 2006, p. 31). It is at the operative level where Kotnik sees the computer's huge potential in geometric modelling beginning to be used.

At the next level of Kotnik's (2006, 2010) digital design model, the parametric level, he argues that the awareness of the architectural designer has moved away from focusing purely on drafting and modelling towards a

mathematical awareness and understanding. At this level, an object is assigned control points and weights to parameterise it within its space. The assigned parameters can be changed at any point during the design process. Kolarevic (2003, p. 17, cited by; Kotnik, 2006, p. 35) points out that it is this flexibility that allows for 'a powerful conception of architectural form by describing a range of possibilities, replacing in the process stable with variable, singularity with multiplicity.'

Kotnik's (2006, 2010) algorithmic level of digital design is reached when the awareness of the underlying algorithmic processes reaches the point where the designer is focused on the actual development of 'computational design logic that is a sequence of algebraic, analytic, and geometric operations for the manipulation of data and its translation into architectural properties' (Kotnik, 2010, p. 9).

At the core of Kotnik's (2006, 2010) Turing-based model of digital design lies Terzidis' (2006) differentiation between computation and computerisation discussed earlier. Terzidis points out that the most common use of computers in architectural design is a 'combination of manually driven design decisions and formally responsive computer applications' (Terzidis, 2006, p. 39). Terzidis (2006) goes on to explain that the problem with this use of the computer is that the designer is not aware of the potential that the computer has and very often does not take advantage of the digital tools available to him, purely because he does not understand that computation is a vital part of the design process. In addition, the designer relies on software that is pre-defined and these software packages (because they are pre-defined) are not able to 'predict the moves, idiosyncrasies, or personality of every designer' (Terzidis, 2006, p. 39). It is, in effect, only when the designer is aware of the computational workings and processes of the computer that the use of the computer in design can be considered digital design.

Kotnik's model supports the notion that the designer's (operator's) awareness of the underlying functions and processes of the computer is

constantly changing and growing. At the descriptive level of Kotnik's Turing-based model of digital design, the designer has no awareness of what the computer is doing. The designer's interaction with the computer is limited to the use of an interface with no conception of how the computer processes information in the visualisation processes, such as drafting and modelling. At the next level, the operational level, although the computer is being used to manipulate objects through geometric operations, the designer is still not aware of the underlying algorithmic operations being processed by the computer. At this level the designer understands and is aware of only the pre-defined transformational tools offered by the computer software. At the third level, the parametric level, the designer's awareness of the actual computational processes of parametric operations has increased and the designer is able to understand and interact with the computer by entering the equations that define the geometry of form building. Finally, on the algorithmic level, the designer is expected to be fully aware of the mathematical operations and the types of transformation rules and functions that the computer is conducting without focusing on the end result that is not pre-defined. This is where 'surprising' or unexpected solutions can evolve.

Kotnik argues that Oxman's descriptive CAD model can be associated with his Turing-based representative model, because both use digital techniques for the design productions, such as drafting and modelling. While Oxman's formation model is sub-divided into the three sub-models of topological formation, associative design model and motion-based formation, Kotnik in his Turing-based model limits this to two sub-models, the operative and the parametric awareness. Oxman's categorisation is based on the notion that the designer's level of interaction with 'an enabling digital technique' is comparable in each of these sub-models and that it differs from the 'explicit representational structure as in the CAD model' (Oxman, 2006, p. 250). Kotnik highlights the divergence between his model at this point because, according to him, there is no comparability possible. For Kotnik the designer's level of awareness of computation is the base of his

classifications and he thus classifies the 'formation models … into the two disjunct levels of operative and parametric awareness' (Kotnik, 2006, p. 39). In Oxman's generative design models, both the grammatical transformative design models and the evolutionary design model are 'characterized by the provision of computational mechanisms for formalized generation processes' (Oxman, 2006, p. 254). However, Kotnik argues that while Oxman's sub-models of grammatical transformative design do deal with form-generation, her sub-models of evolutionary design are not form-making, but rather, deal with optimisation and evaluation.

Kotnik's argument that evolutionary genetic algorithms do not deal with form-making is, in fact, misleading. While evolutionary genetic algorithms can indeed be used for evaluation and optimisation, if an initial form is fed into the genetic algorithm, actual form-creation can be achieved through the processes of regeneration by evaluation and optimisation.

3.2.3 Tectonic Design

Regardless of the actual classification or categorisation of 'digital design', the picture that begins to emerge is that through an increased awareness of the computer and the underlying mathematical, algorithmic processes being carried out by the computer, the designer is able to design the actual design process. Digital design, as Trezidis puts it, 'is a process not a product' (Terzidis, 2006, p. 39). Understanding digital design as a process has, at its core, the notion that the designer 'understands, distinguishes and discerns' the computation that a computer can carry out in the synthesis of design. Digital design, or computational design, differs from the traditional, manual design in the terms, concepts and processes that it includes. What may have seemed 'inconceivable, unpredictable, or simply impossible by a human designer [in manual design] can be explored, implemented, and developed into entirely new design strategies within the digital world' (Terzidis, 2006, p. 39).

In architectural design, researchers and practitioners alike are aware of the potential 'revolutionary' changes that the computer could bring to the practice. The question that arises is whether the increased use of computers in design will eradicate the human essence and creativity from architectural design. If one looks at the use of computers in design, or 'digital design' as a 'process' as Terzidis (2006) does, then one can perceive a poetry or a creativity in the actual process, rather than just in the end product. This notion of poetic expression in both the product and process of making that product comes from tectonics.

The concept of 'tectonics' comes from the Greek word tekton that, according to Kenneth Frampton (1995), describes all art and craft. Martin Heidegger (Heidegger, 1971) states that tekton also encompassing poesis, the making of poetry that, in the larger context of art and craft, we can take to mean the poetic expression of the maker, craftsman or artist. Poesis, seen as the artistic part of making combined with tekne, or the technical realities of building (or formation) both play a big a part in architecture. This is true of both the traditional manual architectural design and 'digital design'. According to Frampton (1995), it was Karl Botticher in his work, *The Tectonic of the Hellenes* who first coined the term tectonics and made the:

> … seminal contribution of distinguishing between the Kernform [the core form] and the Kunstform [the art form]; between the core form of the timber rafters in a Greek temple and the artistic representation of the same elements as petrified beam ends in the triglyphs and metopes of the classical entablature" (Frampton, 1995, p. 4).

Frampton (1995) cites Eduard Sekler as defining tectonics as 'a certain expressivity arising from the static resistance of constructional form in such a way that the resultant expression could not be accounted for in terms of structure and construction alone' (Frampton, 1995, p. 19). Tectonics is an integrated design realization of an aesthetic expression and technical practice. Exploring positions on tectonics from aesthetic analysis to efficient physical structure, Frampton states that, 'everything turns as

much on exactly how something is realized as on an overt manifestation of its form. This is not to deny spatial ingenuity but rather to heighten its character through its precise realization' (Frampton, 1995, p. 26). In this sense, tectonics realises poetic resolution in collaboration with the technical resolution and one can see a certain poetry in the technicalities of digital design.

By examining the various studies and research carried out in digital design, the indication is that the development of CAAD in practice does — and will continue to do so — benefit from the ever-increasing capabilities of the computer as it uses mathematical, deductive bases. In addition to the development of the computer, the design process itself is constantly being improved and developed. As Kalay says, 'through technology, current practices will be displaced, and new ones introduced' (Kalay, 2006, p. 379).

3.3 Space layout planning

In architectural design, planning the layout of the spaces within a building is a core activity. The placement of the various spaces, taking into account the relationships between the different spaces and the distances between them, is an incredibly complex part of any architectural design process. More formally called 'space layout planning', the process of organising the discrete spaces that make up the whole building requires a thorough understanding of the requirements of each space and its relationship, not only to other spaces within the building, but also to the exterior spaces and the surrounding environment. An architectural structure is not a flat 2D representation and the topology and geometry of the building mean that space layout planning is more complex than it would, at first, seem. Space layout planning cannot be done in a linear manner due to complex interdependencies. The planning of the layout of spaces within a building has to be undertaken in a logical process, because it has to deal with a variety of design concerns, ranging from the aesthetic to the cost implications of placing a space in the 'wrong' place within the building.

Apart from the aesthetic concerns associated with the layout of spaces within a building, the architect must also consider the relationships between the spaces and try to place spaces with adjacency requirements together. Distances between spaces can impact not only the functionality of the building, but also the cost of the building (through travel cost between spaces) over its operational life. Per Galle, in his article *An algorithm for exhaustive generation of building floor plans,* notes that 'the combinatorial complexity of most floor plan design problems makes it practically impossible to obtain a systematic knowledge of possible solutions using pencil and paper' (Galle, 1981, p. 813). The use of computational methods in space layout planning has thus been investigated by Galle (among many others) as being a potential method for providing architectural designers with such knowledge.

The process of designing the layout of spaces within a building, or the floor plan, involves two sets of design criteria, the constraints and the *desiderata* (Galle, 1981). The constraints, as pointed out by Galle, can be satisfied or not, there is no subjectivity involved. *Desiderata*, (which can be understood to mean desired or wanted solutions or features), on the other hand, may not be quantifiably satisfied. Galle offers the examples of minimal construction or heating costs, internal traffic, and areas of corridors as quantifiably *desiderata*, and beauty, monumentality or intimacy as unquantifiable *desiderata*. The point Galle (1981) makes is that for a floor plan design solution to be selected, its geometrical arrangements must satisfy all constraints and fulfil as many *desiderata* as possible. Clearly the planning of even the simplest floor plan is incredibly complex and an architect using traditional design methods 'will probably find and contemplate very few solutions' (Galle, 1981, p. 813).

3.3.1 Computer aided space layout planning

Computer-aided space layout planning, variously called 'space allocation', 'automated floor plan generation', 'automated spatial synthesis' or 'quadratic assignment formulation' 'is one of the most interesting, difficult, and controversial areas of computer-aided architectural design' (Kalay,

2004, p. 241). Space allocation, according to Kalay (2004), 'considered by some to be the Holy Grail of CAAD', attempts to use the computer to aid in 'the allocation of spaces (and the activities they house) in a building according to some rational principles (mostly the minimization of distances between spaces that house closely related activities, based on the assumption that such a floor plan would be more 'efficient' for the inhabitants using the building)' (Kalay, 2004, p. 241).

Kalay (2004) notes that several solutions for the computational synthesis of space layout planning have been proposed. In fact, one of the first parts of the architectural design problem that architects and computer scientists attempted to 'subject to computational synthesis' was space allocation (Kalay, 2004, p. 241).

3.3.1.1 Galle's Approaches

Galle (1981, p. 813) identifies five approaches to computational floor plan designing between the 1960s and 1981. These are, in order of degree of automation:

1. On-line machine control and appraisal of the designer's layout proposals
2. Stepwise automatic layout generation interactively guided by manual selection of desirable partial solutions
3. Non-exhaustive automatic generation satisfying given constraints
4. Exhaustive automatic generation satisfying given constraints
5. Automatic generation of optimal or quasi-optimal layouts under given constraints.

The appraisal programs of 'approach no. 1', according to Galle, are favoured by designers who believe that 'human creativity is superior to that of machines in solving real-world problems' (1981, p. 813). These types of programs allow designers to evaluate many more solutions than would be

possible in traditional manual space layout planning. However, the human element (which, depending on one's point of view, can be either a positive or negative point) is still very present in that the candidate solutions are still produced intuitively by the designer and the process of choosing a successful floor plan layout is characterised by an element of human randomness (Galle, 1981).

The automatic layout generation of Galle's 'approach no. 2' is based on graph theory and generates bubble diagrams to represent floor plans. The use of graph theory in space layout planning is discussed in more detail in Section (3.3.2). Although automatic layout generation can be expected to generate more solutions than the appraisal programs (approach 1), 'design becomes no more systematic than the designer's behaviour' (Galle, 1981, p. 814). We assume that this is because the selection process is still guided by the designer. Galle's 'approach no. 3' consists purely of heuristic methods, assisted by the use of analogies (Galle, 1981).

The exhaustive automatic generation of space layout solutions through the satisfaction of given constraints, which Galle lists under 'approach no. 4', was developed by Grason (1968). Grason's approach uses graphs to represent the required adjacencies between the spaces. The adjacencies are represented by the edges of the dual graph, which enables the generation of a dual of the adjacency graph that represents the walls of the spaces (rooms). Although Grason's program was found by Philip Steadman to be able to generate solutions relatively quickly, it was not able to find solutions for problems of more than five rooms (Galle, 1981). According to Galle (1981), Mitchell, Steadman and Liggett, in their 1979 article *Synthesis and optimization of small rectangular floor plans*, outlined a method similar to Grason's with the added step of optimisation at the end of the exhaustive automatic generation process. Galle (1981) explains that Mitchell et al.'s program 'searches a permanent library of topologically distinct 'dissections'' (p. 814) to solve an n-room problem. Galle (1981) goes on to say that in 1977, Earl showed that the algorithm used to generate the library could not be exhaustively used when there were 16 or more rooms

and that Gero, also in 1977, criticised the optimisation method put forward by Mitchell et al. Galle (1981) also tells us that between 1978 and 1979 Bloch, Krishnamurti and Roe 'efficiently generated all dissections with n =< 10, and Stiny's "shape grammars" demonstrated their usefulness for such and other generating tasks' (1981, p. 814). Apart from Grason, Mitchel et al. and Bloch et al.'s various attempts at the exhaustive automatic generation of space layout solutions, Flemming also came up with another method (Galle, 1981). Flemming's two-step method, 'which also satisfies adjacency and dimensional constraints … combines an exhaustive search for certain topologically-distinct equivalence classes of solutions with a linear programming algorithm searching each class for an optimal representative' (Galle, 1981, p. 814).

Galle's (1981) last categorisation, 'approach no. 5', includes methods that are associated with operational research. Examples given by Galle include methods for 'generating a schematic layout minimizing internal traffic [and methods that consider] area minimization' (1981, p. 814). Cost-benefit models that incorporate the analysis of flows of users, heat and loads and the optimal architectural space allocation are also included in approach no. 5. As Galle notes, attempts to reach optimal space allocation are flawed, because it is 'impossible to express all relevant qualities of a floor plan as an objective function' unless the 'notions of numerical utility of imponderables and numerical weighting of incommensurable desiderata are applied' (Galle, 1981, p. 814).

Kalay (2004) has similar views and argues that the results of space allocation programs cannot be 'optimal, or even quasi-optimal … because any arrangement produced by these programs is optimal only in so far as the criterion of distances is concerned' (Kalay, 2004, p. 246). Although attempts have been made to expand the range of the design criteria considered by these programs beyond the distances between spaces, 'their uses have been limited to building types where distances are of paramount importance, like large schools, university campuses, hospitals, and especially warehouses, office buildings, and manufacturing plants' (Kalay,

2004, p. 246). The reason why the use of space allocation programs can be successfully used on buildings such as these is that there are not only a large number of spaces (or activities) within the building, but the actual distance (and thus travel) between the spaces has a measurable impact on the operating costs. The optimisation of the layout is therefore of paramount importance and interactive space allocation programs need to allow the designer to 'add or modify constraints and the layout itself, using the algorithm as both a layout generator and a layout evaluator' (Kalay, 2004, p. 246).

3.3.1.2 Kalay's categorisation of computational approaches to space allocation

Although several solutions (or methods) have been suggested for computationally synthesising space layout design, space allocation works in only a very limited number of architectural design situations.

Kalay (2004) defines the problem of space allocation in general terms as, 'given a set of spaces (or activities) and the desired adjacencies between them, find the layout that minimizes the distances between the spaces that ought to be close to each other' (2004, p. 241). The desired or required distances (proximity) between the spaces can be weighted according to the relative importance of the required proximity. Weighting can be achieved:

> … logically, by considering the relationships between the activities they house, or empirically, by monitoring a building of the same type as the project at hand and recording the number of trips between each pair of spaces (or activities)… the trips can be further weighted according to the different types of users (Kalay, 2004, pp. 241, 242).

The results of the weighting can be depicted in an adjacency matrix, which will list the weight of the connection between each pair of spaces. Over twenty years after Galle (1981) first categorised various methods of computer aided space layout planning into five distinct approaches, Kalay (2004) attempted to make his own categorisations. He distinguishes two general approaches to space allocation, additive and permutation-based.

3.3.1.2.1 Additive space allocation

The additive approach to space allocation, also known as the constructive approach, first deals with the space (or activity) which is most 'connected'. Usually,, this space is placed at the centre of a grid, with the space with the highest adjacency requirements to the first space placed adjacent to it. The third space to be placed on the grid would be the one with the strongest connection to both the first and second space and so on (Kalay, 2004). The algorithm at the heart of this space allocation 'places [the spaces] in such as way that the 'value' of the overall layout (the sum of the weighted distances) is minimized' (Kalay, 2004, p. 242). The produced solution is not however, an architectural floor plan. As Kalay (2004) points out, the results are the generalised mapping of the topological-geometric relative locations of the component spaces, arranged in a manner that privileges adjacency as a design criterion over all other considerations' (p. 242) such as, site, orientation, view and so on.

If desired, site-specific considerations (constraints) can be incorporated into additive space allocation. This can be achieved by giving the grid of the cells (onto which the spaces are placed) 'desirability values of their own' (Kalay, 2004, p. 242). For instance, the grid's proximity to its external environment (roads, boundaries and so on), or the grid's specific orientation towards, for example, *Mekkah* in the case of a mosque design, can be included in the algorithm. As Kalay notes, if the site-specific elements are taken into consideration, each of the spaces that make up the building must 'also include its preferential site attributes, and the placement algorithm must take them into account when searching for optimal locations' (Kalay, 2004, p. 242). Dealing with more than just proximity considerations can potentially lead to conflicts between the proximity and site-specific considerations.

While the additive approach to space allocation searches for optimal locations for spaces, the results (even if only using adjacency considerations) are not 'optimal' layouts, because the first spaces to be laid out will, in all probability, block the optimal placement of subsequent

spaces within the grid (Kalay, 2004). In addition, Kalay (2004) also points out that spaces, such as the toilet, may be placed on the grid first, because they have strong connectivity requirements to all other spaces. Clearly, this is not an ideal situation and while the designer can avoid this by specifying which space should be placed on the grid first, this will not solve the problem of the subsequent mechanical placement of the rest of the spaces. Another problem with the additive approach highlighted by Kalay (2004) is that the method usually produces a concentric layout with subsequent spaces all being placed around the first most highly-connected space. Kalay (2004) notes that, while this approach to space allocation can be successfully used for buildings where natural light in the main spaces is not of great concern, it is not useful for buildings where there is a need for natural light, a rational circulation and other 'macro' considerations' (Kalay, p. 243).

The most significant limitation of the additive approach to space allocation, however, lies in what Kalay (2004) sees as its inflexibility. What is referred to here is that once a space has been placed on the grid, it cannot be moved to another area on the grid. The order in which the spaces are laid out thus dictates the ultimate layout produced by the algorithm, as much as the adjacency requirement matrix.

3.3.1.2.2 Permutational space allocation

The permutational approach to space allocation, also known as the improvement approach, differs from the additive approach in that it enables a designer to change the placing of spaces after the layout has been completed. In this way, the inflexibility of the additive approach can be bypassed. The moving of spaces cannot be made randomly. As Kalay explains it, 'pairs of spaces may exchange their locations, and the value of the layout is recalculated' (Kalay, 2004, p. 243) only if the new layout is better than the original will it be kept, otherwise the first layout will be reselected. This process can be carried out effectively if the number of spaces is small, because all spaces can be systematically tested in their new positions. However, the systematic swapping of all the spaces can only be

useful if all possible permutations are tested (complete enumeration). Because the number of permutations for even a relatively small number of spaces is very large, the process of swapping spaces can, in reality, be completely ineffective (Kalay, 2004). To avoid attempting to test for every permutation (in effect, complete enumeration), spaces that are to be swapped can be chosen at random. This random process can, of course, mean that a number of successful solutions, or the optimal layout solution itself, might be missed. Kalay (2004) notes that the results of random sampling compare favourably with the complete enumeration methods and because the spaces are not inflexibly placed, there is no need to position the most highly-connected space first, as is done in the additive approach. Instead, the permutational approach can begin with a random layout and work towards the most 'satisficing' of 'quasi-optimal' solution through space swapping. Kalay does point out that the initial layout does influence the resultant solutions, saying 'better results may be obtained from different starting arrangements'(Kalay, 2004, p. 244).

The adjacency matrix that the entire space allocation is based on may contain internal conflicts itself; it does not take into consideration the geometrical limitations (such as, the fact that there is a limited area surrounding the spaces and all connected spaces cannot thus be accommodated), a completely optimal solution will never be found (Kalay, 2004). When a layout that is not optimal is reached, and any additional swapping does not improve the solution, a 'local optimum' is reached. This is similar to the 'local optimum' reached in the additive approach when stepwise placement cannot improve upon a solution that is not the optimal layout. According to Kalay (2004), 'although some local optima can be avoided by swapping more than one pair of spaces at a time, the problem is inherent to the approach and cannot be totally avoided' (Kalay, p. 245).

Another limitation of the permutational approach highlighted by Kalay (2004) is that the architect must design the overall shape of the building before the shape allocation is undertaken, because the permutational approach allocates spaces within a predefined contour, rather than the

other way around. This may work successfully if the sizes of the spaces are uniform, but if the spaces are different sizes (which, in reality, is the case in the vast majority of design problems), one of two approaches can be used, either the modular division of the spaces or the magnification of the grid cells. By subdividing each space into modular units, with each unit being treated as a space itself, and with the adjacency matrix reflecting the 'strong connectivity' of the units by assigning them a higher weight, the units that belong to a single space can be kept together through the swapping process. The other method that can be used to overcome the fact that spaces are of varying sizes is to make the grid cells larger than necessary. This allows the spaces to be placed with enough room for swapping smaller spaces with larger spaces. As better layout solutions begin to emerge through swapping, the cell sizes are also decreased so that the spaces are more tightly contained within their cells and are:

> … translated within their respective grid cells to bring them closer to their neighbouring spaces. Such translation may improve the value of the layout, which, combined with the contracting grid cells, will eventually preclude further swapping of the spaces' (Kalay, 2004, p. 246).

The additive and the permutational approaches can, according to Kalay (2004), be used for space allocation in multi-story buildings. Using the additive approach, spaces can be placed not only next to a previously laid space, but can also be placed above it. Using the permutational approach for multi-story space allocation, on the other hand, involves using multiple grids, one for each level of the building. The additional difficult of getting from one space to another over multiple levels must be considered when calculating the value of any space layout produced. 'This can be done by adding an intermediary staircase or elevator space on each floor through which the distance calculation must go, or multiplying the distance by a constant factor' (Kalay, 2004, p. 246). Clearly, a floor plan that has spaces with high adjacency and connectivity on different floors will be unsuccessful.

Kalay (2004) makes it very clear that neither the additive nor the premutational approaches can produce actual floor plans. Any solutions developed are approximations and require the human intervention of the designer, who will take account of many additional design criteria.

3.3.2 Space layout representation using graph theory

Grason (1968) argues that computer-aided space allocation based on any kind of modularised grid representation, whether developed by sequential 'best placements' or random selection, prevents the mapping of the design requirements directly and exclusively into form components. That is, more specifically, location and physical dimension constraints cannot be dealt with independently. Grason (1968) thus suggests that a dual linear graph of the floor plan can be used as an independent topological specification.

The 'graph-theoretic representation of architectural floor plans is as old as graph theory itself' (Kalay, 2004, p. 257). Graph theory was 'almost certainly' first advanced by Leonhard Euler (1735) (Alexanderson, 2006, p. 567). Euler was a great Swiss mathematician of the eighteenth century who solved the puzzle about the 'seven bridges of Königsberg'. Königsberg (which is now Kaliningrad, Russia) is situated on both banks of the River Pregel and also includes the island of Kneiphof that lies at the point in the river where it branches into two. Alexanderson (2006) explains that there were seven bridges over the river and the puzzle that Euler solved was whether a person could negotiate a way through Königsberg so that each of the seven bridges could be crossed only once. The puzzle was thought to be impossible to solve, but Euler demonstrated a mathematical solution to the 'members of the Petersburg Academy on August 26, 1735, and [wrote] up the [solution] the following year under the title 'Solutio Problematis ad Geometriam Situs Pertinentis' (Alexanderson, 2006, p. 567). Biggs, Lloyd and Wilson (1986) in their book *Graph theory 1736–1936* call this beginning of graph theory 'humble, even frivolous', yet they note that 'despite the apparent triviality of [the Königsberg puzzle, Euler] captured the imagination of mathematicians, with the result that graph theory has

become a subject rich in theoretical results of a surprising variety and depth' (Biggs et al., 1986, p. 1).

By the late 1960s and early 1970s, over two hundred years after Euler introduced the concept of graph theory, Grason showed that graph theory could be effectively used in space layout planning. Grason (1968) in his paper *A Dual Linear Graph Representation for Space-Filling Location Problems of the Floor Plan Type*, argues that for computational space allocation a dual graph representation is superior to grid mapping. Grason (1968) supports his argument by showing that dual graphs can be used effectively as both 'requirement diagrams' and 'form diagrams' that allow the designer to map the primary design constraints to independent realisation.

After Grason introduced the use of linear graphs for the representation of floor plans (Grason, 1968; 1970, 1971; Levin, 1964; Rittel, 1968; Teague Jr, 1968), today, almost three hundred years after its 'humble and frivolous' beginnings, graph theory has advanced to support analysis (Wu, Lee, Koh, Aouad & Fu, 2004). Grason's use of graph theory for space layout planning involves the representation of a floor plan in the form of a linear graph. He calls this a 'floor plan graph' and in it the regions, nodes and edges of the linear graph represent the spaces, corners and wall segments of the floor plan respectively. A dual graph representation of a floor plan can be constructed by having each space represented by a node with edges drawn to join the nodes of adjoining spaces. Common convention rules that the edges should be drawn following certain basic rules, (i) edges crossing north-south wall segments are coloured, dotted, and directed from west to east, (ii) edges crossing east-west wall segments are coloured, slashed, and directed from south to north. Dimensioning is made possible by labelling the edges with the length of the wall segment they cross (Grason, 1968, p. 171).

To use a dual graph representation to solve a design problem, the designer would start by outlining the building according to the four outer directed,

slashed edges of the graph. The rest of the structure, spaces (represented by nodes) and their adjacencies (represented by edges) are subsequently filled in, in response to the design constraints, one node after another and one edge after another. The wall segments and spaces of the floor plan graph correspond directly to the edges and nodes of the dual graph. The three basic constraint types can thus take the form of (i) contiguity, (ii) communication and, (iii) length (Grason, 1968, p. 173).

As Ruch (1978) points out, however, not all graphs can be translated into floor plans. To avoid intersecting edges, and thus conflicts in adjacency, only a planar graph can be implanted into the plan as a representation of a 'realizable' floor plan (Grason, 1968; Grason, 1971; Levin, 1964; Ruch, 1978; Teague Jr, 1968). A test for planarity can be carried out and in the event that a graph depicting a set of input adjacency requirements is found to be non-planar, one or more of the adjacency requirements can be changed to make it a planar graph, which can then be used as the basis for a layout plan. The conflicting adjacency requirements that preclude the graph from being a planar graph can be highlighted and listed, with suggested alternative adjacency requirements, by the computer. The designer is then free to choose those requirements that he feels able to relax to transform the graph into a planar graph. The embedding can only take place once the graph has passed the planarity test (Ruch, 1978).

Graph theory is seen to provide a foundation for representing building components by looking at the constituent spaces and applying the relationships between the spaces to produce both 2D and 3D graphs. This means that a better interpretation of the designer's understanding to the design problem and solution can be achieved.

3.4 Generative and Evolutionary design systems

Prior to embarking on an exploration of the generative and evolutionary systems in architectural design, it is worth exploring the algorithms that

are used as the main engine directing the workings of the computer's processing power, because it aids in the design process.

3.4.1 Algorithms

At the heart of all Computer Aided Architectural Design (CAAD) lies an algorithm, working in the background. In fact, as Kotnik points out, 'without exception, using a computer always means to activate an algorithmic procedure as mediator between input and output' (2006, p. 21). CAD systems, according to Terzidis (2006) 'are in essence collections of algorithms each of which addresses a specific graphical design issue' (p. 41). An algorithm is, in effect, the 'mediator between the human mind and the computer's processing power' (Terzidis, 2006, p. 15). Ironically, the vast majority of people who use computers are not consciously aware of the algorithmic processes that make computers do all the wonderful and clever things to which we have grown accustomed. In the context of computer-aided design, Terzidis (2006) points out that 'when a designer uses an algorithm to design, the designer may not be aware, knowledgeable, or conscious of the mechanisms, specifications, or repercussions of the … algorithm' (p. 23). There is no doubt that the development of the hardware of a digital computer is an amazing human feat, however, without the algorithm, which is, in effect, a list of instructions to the hardware, the computer would be nothing more than a box with electrical and mechanical interconnections that could be used as nothing more than a doorstop.

Computations that we now associate with computers are not bound to the physical hardware. Computation in the form of logical, mathematical processes can be traced back to ancient times. The word algorithm itself believed to be derived from the Latin form (*Algoritmi*) of the Arab mathematician Mohammed ibn-Musa Al-Khowarizimi's name. Al-Khowarizimi, who lived during the 8th and 9th centuries, is considered to be one of the greatest mathematicians of all time and is credited as being the founder of algebra. Although algorithms are used for every computer process, they have been used as 'mathematical or logical mechanisms for

resolving practical problems' for many years before the invention of the computer (Terzidis, 2006, p. 15). When the computer was invented, algorithms offered a way in which problems could be fed into the computer for implementation. To be able to enhance and develop the human usage of the computer, it seems clear that human operators need to be more aware of the processes driving our computers. In the context of computer-aided design, as Kotnik (2006) points out, 'what is needed is a conscious consideration of the machine in order to be able to reinvent the design process in architecture and use the computer creatively' (2006, p. 21). Kotnik is not suggesting that the focus should be on the machine itself, but rather, on the processes that can run the machines — in other words, the focus should be on the algorithms that act as a 'language' for the human-to-computer interaction and which 'drive' and direct the computer's processing power.

In the realm of CAD, Terzidis (2006) highlights that 'a user of a CAD system, i.e. a designer, makes use of … algorithms without knowledge of how they work and consequently is unable to determine the full value of their potential' (p. 41). The fact that the designers (computer users) are not the programmers of the algorithms, and may be using algorithms that were programmed to resolve a completely different problem, may mean that completely unexpected results and behaviours are produced. Terzidis (2006, p. 23) argues that:

> … algorithmic tools are abstract, rational and intellectual in nature and therefore related to the human mind [which in turn means that] the output of an algorithm must be associated to a human mind, either the programmer or the designer.

This would suggest that for a designer to be able to gain as much as possible from the use of computers and to be able to direct the outcomes of the algorithms, he should understand and direct the actual processes of the algorithm.

An algorithm, which is a logical and mathematical process, written by a human developer, seeks to 'address a problem in a finite number of steps" (Terzidis, 2006, p. 45). The processes of an algorithm are written specifically to attempt to solve problems. The wonderful flexibility of algorithms is that attempts can be made to solve problems whose target is known, but they can also be used to solve problems whose target cannot be defined. The algorithms thus 'become the means for exploring possible paths that may lead to potential solutions" (Terzidis, 2006, p. 15). Another interesting feature of algorithms is that they can generate other algorithms, 'not only precise, identical, multiple copies of themselves but also structured text (i.e. code) that when executed will behave as an algorithm" (Terzidis, 2006, p. 19). This generative ability, according to Terzidis means that the use of algorithms does not have to be limited to copying, simulating or replacing manual methods of design, but can also be 'studied as methodologies that operate in ways similar, parallel or complementary to that of the human mind" (Terzidis, 2006, p. 20). In addition because algorithms are able to generate novel concepts, new ideas and forms, they will affect the way a human designer thinks and so change his preconceptions and ultimately the way he designs.

Terzidis (2006) calls the development of algorithms that generate other algorithms 'meta-algorithmics', that encompasses not only the creation and generation of the algorithm but also that of the human programmer. Most architects and designers believe that the design is 'conceived, envisioned and processed entirely in the human mind and that the computer is merely a tool for organisation, productivity and presentation' (2006, p. 21). However, as Terzidis (2006) points out, algorithms can generate results 'for which there is no intention or prediction' (p. 21) and can even produce other algorithms that were also not a part of the initial intention of the human programmer. Manuel De Landa (2002) goes so far as to say that 'in a sense evolutionary simulations replace design, since artists can use [simulation] software [which runs genetic operations] to breed new forms rather than specifically design them'. However, he does go on to argue that

'there is a part of the process in which deliberate design is still a crucial component' (p. 1). Thus, while the computer cannot design without some initial input from the human designer, it is able (through processes being run by algorithms) to actually generate novel solutions.

3.4.1.1 Evolutionary algorithms

Nature is constantly changing and evolving and is used as a great source of inspiration for human designers and programmers alike. Individual organisms in nature contribute to the evolving and adapting of the whole population. Individual organisms are made up of both a genotype and a phenotype. The genotype consists of a set of genes (the DNA of the organism) and the phenotype is the fully-developed organism. The life-cycle of any organism can be seen in three phases, reproduction, development and survival. Through the three phases of the lifecycle of the organism, the evolution and adaptation of the population occurs. Janssen (2004) highlights Dawkins' *Universal Darwinism,* which he says 'suggests that the process of evolution can emerge regardless of the medium, be it biological, computational, cognitive, or through some other form' (p. 6).

The concept of evolution has been used in computation in the form of 'Evolutionary Algorithms' (EA), which, according to Janssen (2004), have been very successfully used in many domains and for a wide variety of problem types. Inspired by the processes of inheritance, mutation, selection and the crossover of natural evolution, EAs are, in many ways, analogous to the process of natural evolution. Evolutionary algorithms are used to generate solutions to problems of optimisation through a cyclical process that evolves a population of individuals in a continuous process of manipulation that allows the population to evolve and adapt as a whole (Janssen, 2004). The individuals that make up the overall population can be anything from a solution to a problem, a design that fulfils certain requirements or even a set of parameters in an equation (Janssen, 2004). As Janssen (2004, p. 7) explains, EAs mirror nature:

> The genotype representation is a highly encoded version of the entity being evolved and the phenotype representation is a decoded version of the genotype. The reproduction step creates new genotypes; the development step transforms genotypes into phenotypes; and the survival step allows some phenotypes to survive.

Although EAs 'mirror' and 'mimic' natural evolution, there is a key difference between EAs and natural evolution, EAs require an additional step: the evaluation process that assesses the performance of each individual of the population. In a natural evolution, the assessment of the performance is simply a matter of survival. If an individual (organism) does not die, then it will remain in the population. With EAs, on the other hand, the evaluation of fitness is performed explicitly to assess whether an individual can perform one or more of its objectives (Janssen, 2004).

3.4.1.1.1 Genetic algorithms

Genetic Algorithms (GA), which can be seen as a special form of EA, use heuristics to search and mimic natural evolution. Liu, Tang and Frazer (2002, p. 263) explain the workings of genetic algorithms as being:

> … highly parallel mathematical algorithms that transform populations of individual mathematical objects (typically fixed length binary character strings) into new populations using operations patterned after (1) natural genetic operations such as sexual recombination (crossover) and (2) fitness proportionate reproduction (Darwinian survival of the fittest).

GAs are used to simulate the concept of evolution with computer logic. GAs have a long history of development. They were first presented by John Holland in the 1960s in his investigation of the process of natural selection systems (Holland, 1992). Much like nature, GAs begin with an initial population of individuals which can be generated at random or heuristically (Wong & Chan, 2009). At each step of the evolutionary process, the individuals in the current population are decoded and evaluated according to a predefined quality criterion, (usually referred to as the fitness, or fitness function).

> To form a new population (i.e. the next generation), individuals are selected according to a selection scheme, such as the roulette wheel selection scheme, where individuals are selected with a probability proportion to their relative fitness. This ensures that individuals with greater fitness have a better chance of 'reproducing'" (Wong & Chan, 2009, p. 649)

Genetic algorithms can be used to perform the evolutionary process of crossover to produce a new generation of individuals (or solutions). Crossover is the process through which parts of the 'parents' or genomes (the encoding) of two selected individuals are exchanged. By exchanging parts (usually the stronger or fitter parts) of the parents to the new individuals, 'offspring' or 'children' are, in effect, formed. At the most basic level, after a crossover point is randomly selected, 'substrings' can be exchanged. The crossover operation can be very effectively used in problems that involve the search for new aspects of a search space.

Genetic algorithms are also used to perform the mutation processes found in natural evolution. The mutation operator is processed by the genetic algorithm by 'flipping bits at random, with some small probability' (Wong & Chan, 2009, p. 649). The mutation operation of the GAs randomly samples new points in the search space, and can be used in the attempt to stop convergence to a local optima occurring prematurely, that is, a situation where the solution is clearly not the optimal solution, but no other alternatives present themselves.

Every process or operation of a GA undergoes a vast number of iterations at random, which means that convergence may never be reached. In cases where it is likely that there will be too many possible solutions and thus too many iterations for the GA to process, the designer must specify a point of termination for the GA. This condition for termination can take various forms, ranging from setting a maximum number of iterations or generations to setting an acceptable level of fitness (Wong & Chan, 2009).

The genetic make-up of the second (and all subsequent) generations in the evolutionary process is dependent on the process of selection that decides which part of the population will pass to the next generation. All solutions generated by the GA in the population are evaluated and the selection of the successful solution is made on the basis of fitness. Because the GAs process a vast number of possible solutions, as opposed to just one solution at a time, it is much more likely that an acceptable solution will be found (Kalay, 2004, p. 283).

As Cross (2001) argues, in the practice of design, (unlike the practice of science that requires the results to be validated and confirmed by the method), 'results do not have to be repeatable, and, in most cases, must not be repeated, or copied' (p. 51). The use of genetic algorithms can be seen to respond to this need for varied, 'surprising' and innovative solutions in the realm of architectural design because, although they run set sequences of operations, the results are not repeated. Manuel De Landa (2002), however, points out that for evolutionary results to be truly surprising, the search space of the algorithm needs to be sufficiently rich, because if the designer can easily foresee what forms will be bred the use of genetic algorithm is not being used appropriately.

3.4.2 Generative design systems

Traditionally, the software used in CAAD has been used for drafting, visualising, data checking and the analysis of the specific elements of a building's performance. This can be very useful to the designer to aid in the understanding of their designs and in the development of their knowledge and skills in areas such as geometry (Chase, 2005). As Chase (2005) argues, however, generative design tools, which seek to generate three-dimensional forms, can expand the uses of the CAAD by actually guiding a designer down an 'exploratory' path. Shea, Aish and Gourtovaia note that using performance-driven generative design methods actually enables the computer to become a 'design generator' (Shea, Aish & Gourtovaia, 2005).

Design in a general sense can be seen as the search for solutions to design problems. According to Herr and Kvan (2007), generative design solutions are the result of what they call the search for 'strategies to facilitate' that very search or exploration for solutions through the use of computers as 'variance-producing engines to navigate large solution spaces to achieve unexpected but variable solutions' (p. 61). Computer-aided generative design systems make use of algorithms to generate a host of different solutions from a given set of design goals and constraints. The use of the computer in generating designs can be an invaluable aid to the human designer, because it is able to compute large and complicated sets of numerically formalised dimensions and constraints that a human designer would not be able to process with the same accuracy or speed. However, the use of generative design systems does not make the designer's role redundant because he is still required to evaluate and select the most appropriate or 'satisficing' solution. As Herr and Kvan (2007) point out, some design decisions do need to be made with more understanding of the design context and these will have to be made by the human designer. Thus, various generative design systems offer differing levels of automation and the requirement for human intervention. Even in 'fully automated' generative design systems, the designer does contribute to the design process by defining the constraints of the design problem and inputting the relevant variables into the system. Once the computer's generative process is finished, the designer again intervenes by choosing the successful outcome or deciding on rejecting it and possibly changing the initial variables before re-running the generative processes (Herr & Kvan, 2007). Generative design systems that call for more human intervention can be more responsive, with the designer intervening at pre-defined stages throughout the process.

3.4.2.1 Janssen's categorisation of generative design approaches

Janssen identifies three main approaches to generative design that can be used within the developmental step of an evolutionary system to create

alternative design models. While Janssen (2004) highlights three approaches, he acknowledges that other approaches and techniques to generative design exist. The three approaches that Janssen (2004) focuses on are the 'parametric approach', the 'combinatorial approach' and the 'substitution approach'. The combinatorial and substitution approaches are, according to Janssen (2004), more flexible than the parametric approach. However, a combination of all three of these approaches can be used in a generative evolutionary design system.

3.4.2.1.1 Parametric approach

The parametric approach to generative design systems allows the generation of a range of forms by first assigning parameters to either a model or a procedure and then varying some of these parameters. As Janssen (2004) points out the parametric approach does not only cover approaches where only dimensional parameters are varied, but actually encompasses rational modelling, variational design, constraint-based design and so on. Janssen (2004) explains that, viewed from a wider perspective, the parametric approach can be seen as the 'varying of constraints, where dimensional parameters are just one type of constraint' (p. 57). The parametric approach offers a great deal of control to the designer over the generation process. However the approach is limiting because the form itself is not very variable (Janssen, 2004).

Janssen (2004) cites Monedero as highlighting two techniques of parametric modelling, the 'variational technique' and the 'history-based technique'. The variational technique is used in parametric modelling systems and must have a predefined model of the form to be generated. Using this technique, the constraints or parameters (including the dimensional parameters) of a model of the form that is going to be generated are defined as equations, which are subsequently solved 'simultaneously by a constraint solver' (Janssen, 2004, p. 57). The history-based technique, on the other hand, generates forms incrementally through a series of operations that require certain data values. The manipulation of the generated form can be carried out by the operations or the data that is

entered for each operation. This history technique generates forms through referencing the sequence of operations used to generate a form, while the variation technique does not make any reference to past operations.

3.4.2.1.2 Combinatorial approach

The combinatorial approach, which generates forms by combining a predefined set of elements, is, perhaps, according to Janssen (2004), the most general kind of approach to form generation. The combinatorial approach generates forms by combining and assembling elements (or components) of different types, although it is also possible that the components are of just one type. Janssen (2004) identifies two combinatorial techniques, the first using an algebra technique and the second using a template technique.

Using an algebra technique, a set of components or elements are defined that include not only the element types, but also include a set of operations for manipulating their position (Janssen, 2004). The operations are then used to assemble components. Janssen (2004) cites as describing the elements and operators as an 'algebra', which is why this is called the algebra technique. Janssen (2004) notes that the algebra technique is very flexible because of the possible assemblies of elements are not very restricted and it is widely used in most standard CAD software. However, when using this technique, the types of forms that are generated are difficult to control. Using the template technique, on the other hand, allows for much more control, but the variability of the generated forms is reduced. The template technique allows elements to be inserted into a set of locations or place-holders in a pre-defined organisational template. Each place-holder has a list of associated possible elements that can be inserted at that location in the template and by inserting elements into the place-holders a range of forms can be generated.

3.4.2.1.3 Substitution approach

The substitution approach generates forms by beginning with a seed form that is provided. Modification and manipulation can be carried out by

repeatedly substituting parts or components of that seed form into the new parts to generate a new form. The manipulation of the seed form is carried out by a set of transition rules that change or modify elements of the form. The transition rules are, according to Janssen (2004), *if-then* type rules, which specify 'that *if* a certain configuration occurs … *then* [*the] configuration must be replaced by a new configuration*' (p. 65). The transition rules can be applied a number of times during the generative process and can also be applied to the generated forms they produce. Not only can a variety of forms be generated by modifying the seed form but alternative forms can also be generated by modifying the transition rules, modifying the total number of steps carried out by the rules or modifying which transition rules to apply. The substitution approach can be used to generate very complex forms using only a small number of rules. Nevertheless, as Janssen (2004) points out, the approach is both unpredictable and difficult to control.

Janssen (2004) claims the substitution approach to generative design can be achieved by using one of two techniques, generating forms by using a grid or generating forms by analysing the shapes of the components. The 'grid-based' substitution technique uses a pre-defined cellular grid into which the substitution of the various parts of the form can be made. On the other hand, when using the 'shape-based' technique, the substitutions are 'performed based on the geometry of the individual shapes' (p. 56). The design space, rather than being defined by a grid, is a continuous space. Growth is thus allowed in this continuous space where new shapes substitute existing ones (Janssen, 2004). An example of the 'shape-based' technique is the use of shape grammars (discussed in Section 3.2.2.1 Kalay's methods), which Agarwal, Cagan and Constantine (1999) argue are especially useful in designing forms 'that are differentiated primarily on the basis of form yet driven by function' (p. 253).

3.4.3 Evolutionary design systems

Evolutionary architecture is the study and practice of architectural form-generating processes that use the model of nature and the morphogenesis

found in nature as its inspiration. The term 'evolutionary architecture' (Frazer, 1995; Frazer, 1974; Frazer & Connor, 1979) was used to describe the efforts to emulate 'the awesome creative power of natural evolution by creating virtual architectural models which respond to changing environments' (Frazer, 1995, p. 9). The ultimate aim of evolutionary design is for the built environment to have the 'symbiotic behaviour' and 'metabolic balance' that is characteristic of the natural environment (Frazer, 1995). As Herr and Kvan (2007) explain it, 'in evolutionary design approaches, the design process is organised as a cyclic process that generates increasingly appropriate solutions by way of repeated selection at every design cycle' (p. 61).

Evolutionary architecture, unlike traditional architectural design systems, incorporates more than just the idea of spaces and forms, but includes a set of generative rules stated in a genetic language that produces instructions for the generation of those spaces and forms (Frazer, 1995). The generative rules used to evolve design solutions in evolutionary architectural systems are evolutionary algorithms (discussed in Section 3.4.1.1). These evolutionary algorithms seek to mimic the adaptive evolutionary processes of nature by applying the notions of natural selection, mutation and recombination to computer systems that are designed to generate architectural spaces and forms. The evolutionary algorithms are used to evolve a population of configurations in a process analogous to natural selection which uses objective functions, called 'fitness functions', to evaluate the evolved solutions. The evaluation and selection of the generated design solutions is carried out by simulating the environment of the designed entity, which is thus developed out of a series of evolutionary steps based on the 'artificial' selection of solutions that best respond to pre-defined fitness function criteria, with the removal of poorly performing designs. Gu, Tang and Frazer explain this 'artificial selection' as the 'selective breeding carried out by humans to produce a desired evolutionary response' (2006, p. 224).

As Gu, Tang and Frazer (2006) point out, evolutionary systems have been implemented in various fields of design, including graphic, media, art and industrial design and the main problem encountered has been that 'convergent mechanisms of standard evolutionary algorithms are difficult to achieve' (p. 224). The genetic algorithms (a kind of evolutionary algorithm discussed in Section 3.4.1.1.1) running the evolutionary systems use selection pressure (described by Gu et al. as 'an index of the unevenness of selection) and population size to 'control the diversity in a population' (Gu et al., 2006, p. 225). This is important, because interbreeding in a small population or a high selection pressure can result in premature convergence.

Although genetic algorithms can be developed specifically for minimising the selection pressure through means including 'proportionate selection', 'tournament selection' and 'truncation selection, all these methods rely on the objective fitness functions being very well defined. As it is proposed in this research, (discussed in Section 5.1.3) the limited population size, on the other hand, results from the human designer selecting the successful solutions from the population. This need for human intervention restricts the population size, because only a limited number of individual solutions can be displayed to the designer by the computer. Again, well-defined fitness functions can be used to minimise the need for the human designer to intervene in the selection process. Additionally, a hierarchical evolutionary system, with optimisation cycles at each level of the system, could offer a way of allowing selection from larger populations of evolved solutions in each level. Premature convergence of the overall solution is less likely, because the population at each level is relatively unrestricted, deals with fewer constraints and limitations, and the selection pressure at each level would then be the case if all design constraints and limitations of the design problem had to be tackled at the same time. Once a population of solutions is chosen in one level, the evolutionary process can begin in the next level starting with a relatively unrestricted population space and relatively un-pressured selection.

Sun (2002) proposed a generative and evolutionary design system in which he developed a design schema of 'formatives'. A 'formative' defines a set of 'rudiments' and their relations along with the generative rules governing the generating process. A 'rudiment' defines a set of 'primitives' which are the geometrical forms and structure defining functional components with related design knowledge. This configuration of primitives, rudiments and formatives adopts CAD modelling representation, parametric design and feature-based representation methods. During the system implementation, only the representation of the functional component classes, i. e. the rudiments which constitute a formative, were encoded as a structured code script for the GA program. The configuration rules were generated and modified manually.

Sun's system was developed to support product design and was tested and implemented on a mobile phone design. The binary structured code script of Sun's system allows for a degree of flexibility and reusability of the system in designing different products.

3.4.3.1 The five key evolutionary design systems

The potential use of genetic design systems, as mentioned earlier, has been researched in many fields and, consequently, there are a number of systems and approaches that have been designed. Janssen (2004) outlines five key evolutionary design systems and approaches. These are the 'Genetic Algorithm for Design Optimization' (GADO), 'Generative System' (GS), 'Genetic Algorithm Designer' (GADES), the 'concept-seeding' approach and the 'epigenetic design' approach.

3.4.3.1.1 Genetic Algorithm for Design Optimisation (GADO)

The Genetic Algorithm for Design Optimisation (GADO) developed by Khaled Rasheed (1998) uses a genetic algorithm 'for continuous design-space optimization that uses new GA operators and strategies tailored to the structure and properties of engineering design domains' (Rasheed, 1998, p. ii). GADO is an evolutionary design system that focuses on the

parametric design phase, where 'more detailed decisions about numerical aspects of the design' are made, rather than on the structural design phase, which 'involves making high level decisions about the overall shape of the artefact' (Rasheed, 1998, p. 1). GADO, as Janssen (2004) explains it, is a parametric evolutionary design system that can be used to optimise engineering design by using a 'parallel master-slave model and a steady-state asynchronous evolution mode' (p. 140). The GADO system uses and maintains an initial population, with specially-designed operators for crossover and mutation being applied to this population. Because designs are created through these operations working on the initial population, they are added to the population. 'A module has also been developed to ensure that diversity in the population is maintained' (Janssen, 2004, p. 140).

3.4.3.1.2 Generative System (GS)

According to Janssen (2004), the Generative System (GS), developed by Caladas, is, like GADO, also a parametric evolutionary design system. The GS focuses on the environmental performance of the building, rather than the engineering aspects. Caladas' GS seeks to develop designs that can then be simulated using the DOE-2 application. For instance, the DOE-2 application was used by the GS for lighting and thermal calculations and, according to Janssen (2004) sometimes Pareto multi-criteria optimisations were also used. With the GS, the 'systems follows the general synchronous evolutionary architecture, and uses standard rules and representations' (Janssen, 2004, p. 140).

3.4.3.1.3 Genetic Algorithm Designer (GADES)

The Genetic Algorithm Designer (GADES) is a generative evolutionary system developed by Bentley (Janssen, 2004). Janssen (2004) cites Bentley as claiming that GADES can be used for a variety of design types. The GADES system 'follows the general synchronous evolutionary architecture, but uses specifically developed rules and representations' (Janssen, 2004, p. 123). The evaluation phases or routines used by GADES are also

customised and thus the system is 'supposed to be highly generic and applicable to any design domain' (Janssen, 2004, p. 123)

3.4.3.1.4 The concept-seeding approach

As Janssen (2004) points out, John Frazer takes a completely different approach to generative evolutionary design in relation to the role of the designer. Frazer, Janssen (2004) claims, focused first on the generative systems and then enhanced them with evolutionary capabilities. In Frazer's approach, the designer captures and codifies his design ideas, which are then entered into a computer program that can generate new designs that still embody the original design ideas (Janssen, 2004). This approach of Frazer's is described as the concept-seeding approach, because it allows the designer to capture the design idea as a seed that can subsequently be manipulated and developed in response to specific problems, and can also be evaluated by the evolutionary system. The capturing of the seed is achieved by creating a set of rules and representations that encompass the designer's design concept or ideas. The seed of the concept can include any design considerations, such as the formal, structural, constructional, aesthetic or indeed any other consideration.

3.4.3.1.5 The epigenetic design approach

The epigenetic design approach extends the role of the environment out of the evaluation phase of the design and allows designs to actually be generated in response to the design environment. As Janssen (2004) points out, a number of evolutionary design systems have the capability of using encoded environmental information in the evaluation step. This encoded environmental information can take the form of encoded design constraints or encoded design context. The epigenetic design approach, on the other hand, uses this environmental information in the developmental stages of the design process that, in reality means that the ultimate successful design is even more adapted to its environment.

4 Conceptual Design Method

This chapter introduces the Conceptual Design Method developed as a result of the Exploratory part of the 'conceptualisation' stage of the research project. The first part of the 'conceptualisation' stage of the research is the research Clarification, including a literature review and the identification of the most relevant areas for further development, which are discussed in Chapter 3 Literature Review. This chapter results from a more in-depth exploratory review of the areas covered during the first 'clarification' part and aims to identify factors influencing the preliminary criteria and to develop a reference and impact model, including success criteria and measureable success criteria along with key factors that would lead to improvements. This Exploratory review culminates in the Conceptual Design Method and moves the research project onto the next 'developmental' stage.

4.1 The Reference and Impact Model

4.2 The developed reference and impact criteria (Appendix B

The developed reference and impact model, with key factors, successful criteria and the measurable successful criteria), establishes the 'Design Schema', the 'Number of Variants Considered' and the 'Multi-Level Evaluation' as the three main Key Factors that would lead improvement of the existing situation. The Reference and Impact Model also indicates that the 'Early Generation of Successful Designs' is the main Successful Criteria and the 'Quality of Evaluation', the 'Quality of Generation' and the 'Cost reduction' are the Measurable Successful Criteria.

The Conceptual Design Method discussed in this chapter results from an in-depth review of the current research into the main areas of interest and indicates ways in which this current knowledge can be further developed and used. Although many of the underlying theories and systems already exist, this Conceptual Design Method offers a new way in which the

advantages of several design systems can be brought together in an effective design method. In most instances, this involves not merely the highlighting of the strengths and advantages of existing research, but also the actual manipulation and development of certain aspects to develop a new direction in research into CAAD.

4.3 Introduction

Similar to all computer technology in this day and age, it is widely believed that the full potential of Computer Aided Design (CAD) systems has not yet been reached (Boddy, Rezgui, Cooper & Wetherill, 2007). With computer systems and technology developing at a rapid rate, this is not surprising; but in the case of CAD, it is not only the rapid changes in technology that is the main reason for believing that there is enormous scope for improvement. It is believed by many researchers that the gap between the unconscious flow of creative thought of the designer and the conscious thought required to operate a computer poses an almost unsurmountable obstacle to being able to adapt the CAD systems to conventional design methods (Liddament, 1999). Currently, the vast majority of instances where computers are used in the design process involve drafting and visualisation rather than design generation itself (Holness, 2006).

There have nevertheless been some promising developments within the field of CAAD systems with the conceptual design method proposed by this research attempting to amalgamate the benefits of a number of systems into one. The research into the conceptual design method proposed is based primarily around the generative and evolutionary design systems.

Existing generative design systems generate design solutions from a ‘seed’, out of which a form is grown according to a set of rules governing the growth process (Frazer, 1974; Frazer & Connor, 1979). Evolutionary design systems work in a similar way, but use environmental considerations to evolve designs from the ‘genetic code’ of an initial design concept. Much like the evolutionary processes in nature, design solutions are evolved in

response to the environment through 'natural selection' (Frazer, 1995, p. 65).

In addition to the generative and evolutionary design systems, current research in performance-based design is also used for the development of the conceptual design method. In recent research, performance-based design has been highlighted as a comprehensive design approach that can greatly facilitate the search for sustainable design solutions (Kolarevic & Malkawi, 2005, p. 195). At the heart of the performance-based design is the simulation of performance targets for the building that ultimately leads to the development of design solutions in response to their simulated environments (Kolarevic & Malkawi, 2005, p. 197).

Figure 4-1 The conceptual design method, is a graphical representation of the conceptual design method. A number of key stages can be identified.

At the first stage of the conceptual design method, the designer captures the initial design concept. This initial design idea is influenced by the designer's preconceptions in addition to the constraints of the design problem being tackled. Both the internal constraints of the design situation, and the external constraints, will play a large part in the formulation of the design concept. The quality of the problem definition therefore is essential to the designer's interpretation of the design problem and subsequently to the conceiving of the design schema.

The second stage of the conceptual design method involves conceiving the design schema that has been identified earlier as a Key Factor. The design schema is an encoding of the designer's creativity within the design process when he/she tackles the design problem at hand. The schema will thus necessarily also include the designer's understanding of the design problem as a whole, in addition to an understanding of the finer detail involved (all the elements and components that constitute the building), both of which are influenced by the designer's preconceptions.

The third stage of the conceptual design method involves the encoding of the design schema, together with the internal and external design constraints (which represent the environments of the design problem), into the generative evolutionary design system.

The evolutionary algorithmic design system, set-up in a hierarchical manner, represents the fourth stage of the conceptual design method. The number of variants considered and the multi-level evaluation, as the other Key Factors, are accounted for within the evolutionary design system. Here, the various levels of the system process different sets of variants according to the design schema and in response to the design environments, represented by the design constraints as fitness functions for the genetic algorithm.

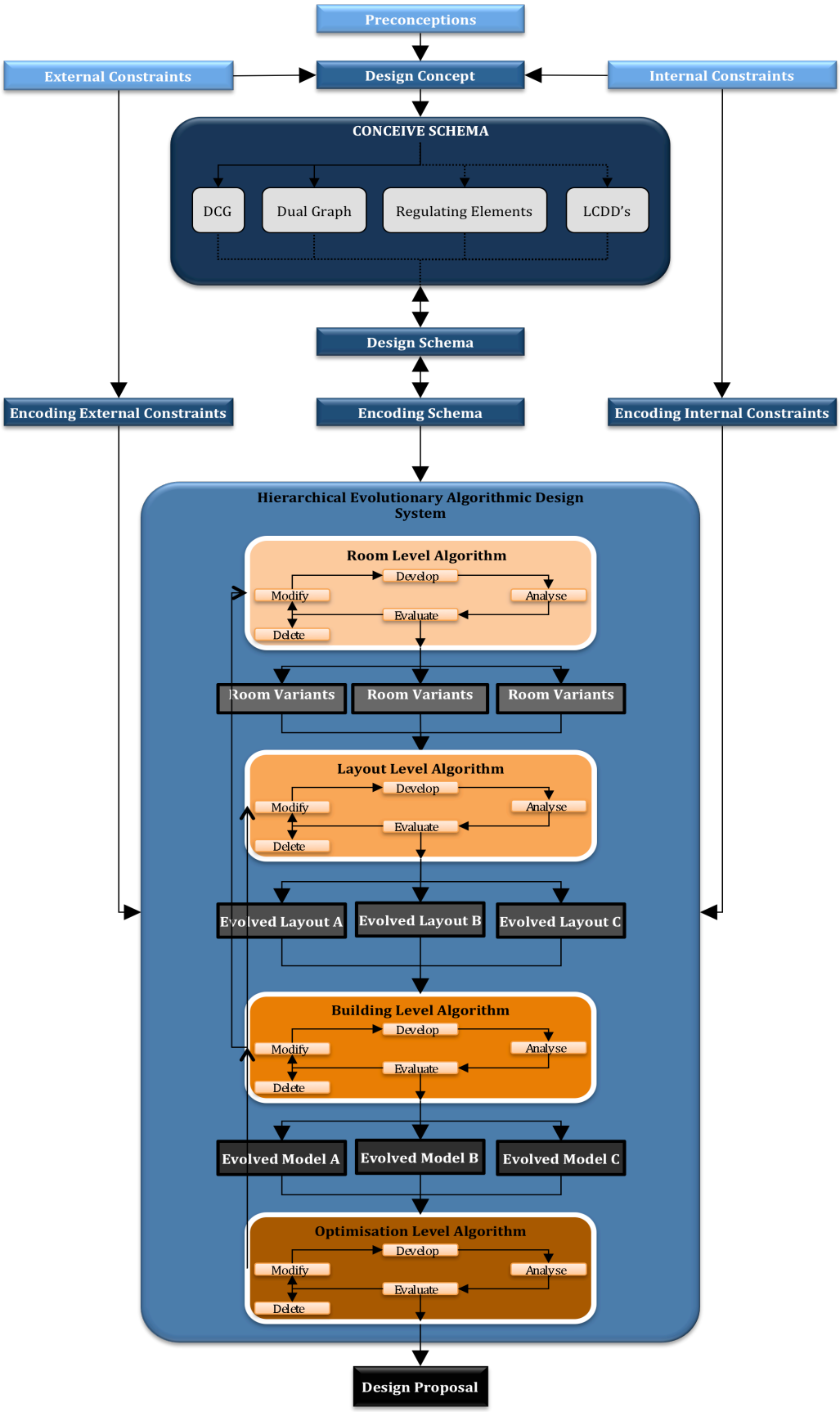


Figure 4-1 The conceptual design method

4.4 Design context

The design context incorporates every known aspect of the design problem. All the project stakeholders, including the clients, users, legislators and the designer, assist in generating the design problem. The client's needs and requirements obviously form a large part of the initial design problem. The user, who may not necessarily be the client, might very well have different needs that will contribute to additional aspects of the design problem. In a similar way, legislation, such as building codes and regulations, will also greatly affect the design problem. Most frequently, this will be through constraints within which the design must be tackled. The designer himself is a generator of the design problem through his subjective and objective interpretation of the requirements of the design.

Given the multitude and variety of generators of the design problem, the design problem itself is very often not immediately obvious and cannot be definitively formulated. More often than not, many aspects of the design problem come to light only when the problem-solving process has begun, and equally often, different problems come to the fore throughout the process. As Lawson points out, some aspects of the design problem may never actually emerge as explicit parts of the problem (Lawson, 2006). As Rowe (1998) in his book *Design Thinking,* puts it, the 'problem space' is made up of 'knowledge states', some of which might be solutions to the problem. Once problem-solving activities have begun, some of the original 'knowledge states' may actually produce new 'knowledge states'. In effect, the 'problem space' itself can change through problem-solving. Thus, the design problem cannot be seen as an unchanging distinct set of requirements and constraints.

An attempt at defining the design problem is made in the form of a design brief. Unfortunately, the design brief is often received from a client who may well know what the project's requirements are, but is unable to formalize them into a design brief. To complicate matters further, in some cases clients are unable to provide the designer with a brief at all, because they do not know exactly what they require. The brief, and thus the design

problem, therefore becomes dynamic, because whether it originates from the client or whether the designer has to formulate the brief himself, it develops within the design process as the designer learns more about the problem through his attempts to solve it. Darke (1979) quotes MacCormac as saying, '… you can't start with a brief and [then] design, you have to start designing and briefing simultaneously, because the two activities are completely interrelated' (Darke, 1979, p. 42). Any 'design problem' that emerges through the design process will, in addition, be based on the subjective interpretations of the client, the user and designer. Thus, no design brief can ever be expected to be an entirely objective interpretation of the design problem. Most design problems seek to satisfy multiple functions which are both overlapping and interacting, thus design solutions need to respond to a multitude of requirements and therefore tend to be organised hierarchically (Lawson, 2006).

4.4.1 Design constraints

Bertel, Freksa and Vrachliotis (2004) and Lawson (2006), describe constraints as the 'result from required or desired relationships between two or more elements' of the design problem (Bertel et al., 2004, p. 267). Essentially, the design problem is generated through the requirements and limitations imposed by the stakeholders of the project on the design solution. This imposition of constraints is not necessarily overt or explicit and the extent varies for each constraint that has to be adhered. For example, as Lawson points out, the constraints imposed by the designer himself will be much more flexible than those imposed by the legislator. Certain constraints imposed by the client or the designer, for instance, can be re-appraised if conflicts arise, while constraints imposed by legislators, such as those dealing with safety issues, would need to be rigidly adhered to (Lawson, 2006). Clearly, design constraints are generated by many different stakeholders and affect the design solution to greater or lesser degrees.

Tang and van Vliet (2009), in their paper on modelling constraints in software architecture and design reasoning, group constraints into four

categories, 'requirement related constraints', 'quality requirement related constraints', 'contextual constraints and solution-related constraints'. Although Tang and van Vlient's categories of constraints relate to software architecture, on closer inspection we see that their categorisation could also be adopted to group constraints in building design. Tang and van Vliet see requirement-related constraints as those constraints generated by the functional requirements of the design. This is as applicable in building design as it is in software architecture. Similarly, quality requirement-related constraints are those constraints generated by the quality requirements of the design. Contextual constraints are the constraints generated by factors that will affect the environment for constructing a solution. Examples are: the project context such as the costs and schedule of the project; the technology context, such as what technology platforms, in the case of software architecture, and what technology and building systems, in the case of building design, are authorised for use. Finally, the solution-related constraints are those constraints generated during the design process that, again, are equally relevant to both software architecture design and building design.

Lawson and others take a simpler route and distinguish two types of constraints, 'External Constraints' and 'Internal Constraints'. In his book on *How Designers Think*, Lawson defines the external and internal constraints faced by a designer. The external constraints are the unchangeable parts of the design problem that are not affected by the design method and cannot be changed without creating a further problem. For instance, the site (which includes location, orientation and neighbouring built environments), weather, building regulations and the laws of physics are all prime examples. The internal constraints, on the other hand, include factors such as the functional expression or modes of fabrication and have some level of flexibility. In other words, the internal constraints can be re-assessed and changed without creating any further major problems (Lawson, 2006). The importance given to each constraint and the extent to

which it is allowed to impact the final design solution varies throughout the design process and is determined by the designer.

The design constraints are imposed by the various stakeholders for the building to fulfil, as satisfactorily as possible, all the functions and requirements expected of it (Lawson, 2006, p. 100). A great deal of research into the functions of constraints has been carried out to improve the understanding of the nature of the design problems. Lawson (2006) conducted extensive research that culminated in his development of a model identifying four general functions of constraints, 'radical', 'practical', 'formal' and 'symbolic'. The radical constraints, according to Lawson, comprised issues related to the primary purpose of the building. They impact design decisions both greatly and very early in the design process, because they are the reason for having the design in the first place. As suggested by their name, the practical constraints are related to the practical reality of production. Issues, such as the technological aspects of building erection and the technical performance of the building during its life, are considered as practical constraints. Formal constraints incorporate the visual aspects of the design, such as the proportions, form, colour and texture. Finally, the symbolic constraints include all aspects of style, taste and any interpretation of what the design seeks to symbolise. Because the various functions of constraints can be generated by any one or more of the generators they can, and often do, overlap with each other (Lawson, 2006).

Being able to categorise constraints as either internal, external, radical, practical, formal or symbolic does not, however, suggest that there is any particular way in which a designer does (or even should) tackle the constraints he faces. Lawson (2006, p. 108) points out that it is not clear whether any particular constraints should be tackled first, whether any constraints are critical for determining the design and its success or indeed, whether different designers focus more on different types of constraints. What is clear is that each designer is faced with a wide range of varied constraints, many of which conflict with each other. Bertel et al. (2004) cite Carrara, Yehunda and Novembri as suggesting that 'a prioritisation of goals,

reflecting a descending order of preferences, may be imposed by the designer or by the client, [indicating] which combination of performance criteria the designer should attempt to accomplish first' (Bertel et al., 2004, p. 267). It is this prioritisation of the design constraints that aids the designer as he seeks to find the most satisfactory solution to a potentially daunting multitude of design problems. In his attempt to fulfil the project's goals as effectively and as fully as possible, the designer prioritises the project's constraints in a unique and personal way. Faced with the same set of design constraints, different designers would, in all probability, prioritise the constraints differently, because prioritisation is highly influenced by each designer's preconceptions.

4.5 Designers' preconceptions

A number of researchers argue that architects have preconceptions as they design. It is undeniable that no designer is able to approach a design problem with a completely fresh and unbiased mind. The preconceptions a designer has are essentially a set of personal beliefs, values, attitudes, philosophy, education and past experience that a designer brings to each design problem. The preconceived ideas and ideals a designer has about the practice of design are developed throughout his design career. Preconceptions have, in the past, been seen in a negative light, as a hindrance to the design process. The design method propagated in the 1960's for instance was based on a systematic, scientific and rational approach, which aimed to exclude all forms of subjectivity and thus by extension sought to discourage preconceptions of any kind. In the second generation of the design movement, Broadbent modified the 1960s design process, by including the designer's 'preconceptions' in the synthesis stage of the process (Broadbent, 1988). Janssen however points out that while there are indeed what he calls 'limiting preconceptions' which 'restrict the freedom of the designer', there are also what he calls 'enabling preconceptions' which actually 'give the designer greater freedom' and encourage more creative problem solving (Janssen, 2004, p. 167). Janssen rightly highlights the fact that the ideal would be for architects to nurture

and hold on to their 'enabling preconceptions' and abandon their 'limiting preconceptions'. Unfortunately it is difficult, and in some cases impossible, to know (even for the designer himself) which preconceptions are limiting and which are enabling. This may also vary from one design problem to another. What is clear is that preconceptions, whether they are limiting or enabling, continue to play an essential role in the creative aspects of the design process (Janssen, 2004).

The designer's preconceptions can be placed into three categories, the guiding principles, the primary generators and the designer's schema.

4.5.1 Guiding principles

Lawson calls a designer's personal preconceptions 'guiding principles' (Lawson, 2006), but they are also referred to as the designer's 'paradigmatic stance' (Broadbent, 1988) and 'enabling prejudices' (Rowe, 1998). The guiding principles reflect the philosophical beliefs, cultural values and background of the designer. They tend to reflect the designer's personal expression, influenced by his skills, of the importance of various design constraints and ultimately help to formalise his unique individual stance. The designer's paradigmatic stance is evolved and developed throughout the designer's practicing career (Janssen, 2004).

The designer's preconceptions are a set of principles, as opposed to just one principle, that the architect adopts. Having said that, one particular guiding principle might well dominate the designer's stance. The set of the designer's guiding principles arises in response to the different generators and/or functions of the design constraints. The client and the users, as design constraint generators, together with the practical, radical, formal, and symbolic functions of the design constraints, can all influence a designer's guiding principles (Lawson, 2006).

Client involvement in the design process is seen by different designers at opposite ends of a spectrum — while some designers encourage client participation, others consider it to be a hindrance in the design process. In

some cases, a designer can have such a strong opinion on the subject of client involvement that it can become a guiding principle. User involvement in the design process may not be possible, because the client may not necessarily be the user and the designer may not have direct access to the user. Lawson (2006) cites Habraken as indicating that given the opportunity, the designer would usually prefer to get direct feedback from the users, rather than having to try to look at the design problem from the perspective of the users. The practical function of the constraints, which involves the technology of building erections and the mode and material of fabrication, provides the designer with a strong base for developing guiding principles. The expression of the engineering technology through the materiality and structural elements are reflected in some designers' guiding principles. The radical constraints reflect the essential reason for the building being designed and will thus, inevitably, be the centre of the designer's attention. The radical constraints offer a typological guiding principle for designers. Formal constraints can be observed through geometric and proportional rules, which have come to the frontline recently through the use of computers in design. The use of geometry and shape offers a powerful source for guiding principles to designers. Symbolic constraints, as a source of guiding principles, provide a wide area of opportunity for the designer's creative expressions (Lawson, 2006).

The extent to which a designer's guiding principles affect the design process varies. For some designers, they represent a solid design philosophy that the designer allows to take an important part in the design process, while for others, the guiding principles are not formalised and affect the design process in a much more subtle way. Each produced design is not only a solution to the design problem, but to a lesser or greater extent, a further development in the designer's theoretical notions of the design process itself. The guiding principles contribute to the design process by influencing the capture of the design concept.

4.5.2 The primary generators

'Design is seen as a process of 'variety reduction', with a very large number of potential solutions reduced by external constraints and the designer's own cognitive structure' (Darke, 1979, p. 38). This 'variety reduction' begins when the designer sets about trying to formulate the design problem. The formulation of the design problem involves the framing of the design situation by logically setting boundaries for the problem and choosing particular issues and any associations that require attention (Schön, 1988). In a further effort at reducing variety, the designer adopts an initial design concept or idea, which allows him to concentrate his solution finding activities around a particular chosen concept.

The initial design idea, according to Rowe (1998), so greatly directs the way the problem-solving activity is carried out, that even if problems occur with the design idea, the designer does not abandon the initial idea in search of a new one, but instead, will try to work out the problem associated with the initial idea. Cross (2004) agrees that designers do not easily give up their initial design ideas, even though they change their goals and constraints throughout the design process.

Various scholars attribute the formation of the design solution to the designer's initial design ideas or concepts. The 'design idea' or 'design concept', also referred to as the 'primary generators' (Darke, 1979), 'organizing principles' (Rowe, 1998), 'concept or parti' (Lawson, 2006) and 'generative concept' (Frazer, 1995; Frazer, 1974; Frazer & Connor, 1979), reflects the designer's subjective perception of a particular design process. It is the designer's interpretation of the design constraints, as prioritised by the importance of the project's goals and objectives together with the preconceptions of the designer, that combines to form the design idea.

Several scholars have studied design principles, from the conventional paper-based architectural design practice and its derived digital version through to the digital architectural design practice. According to Janssen, the initial design idea can be sub-categorised into two types, 'design idea as

design heuristics', and 'design idea as a resolution of constraints' (Janssen, 2004, pp. 51–52). Rowe (1998) suggests that the design can result from heuristic methods geared towards solution-finding. The heuristics employed by a designer can be relatively informal and based, to a large extent, on personal experience.

According to Rowe (1998), design constraints can be approached heuristically, based on the information that they provide. Rowe (1998) presented five forms of heuristics that a designer may use in the search for a design solution, (i) 'Anthropometric Analogies', that use the human form as an inspiration, (ii) 'Literal Analogies', that use established and readily recognised shapes and forms, (iii) 'Environmental Relations', that relate to the building's performance within its environment, (iv) 'Typologies', that use established design solutions and, (v) 'Formal Language', that uses the formal, accepted solution as a guide. Rowe points out that because the heuristics used in the design process can be created for a particular use and then can be discarded, they can be both informal and have a relatively short lifespan. 'Although this is especially true of problem-oriented constraints, it can also happen with an autonomous constraint in the form of a 'wild idea' that comes to the fore, is useful for a moment, and is then forgotten.' (Rowe, 1998, p. 91).

The use of computationally-based processes for form finding, as opposed to form making, introduces a new source of design ideas. Kolarevic (2003) highlighted six architectural computational concepts for the digital morphogenesis of form origination and transformation. Kolarevic's six forms of morphogenesis are, (i) Parametric Design, where a design is configured using its parameters rather than using its form, (ii) Dynamic and Field of Force, where the animation of the interaction between space, time and force is used to produce dynamic forms, (iii) Datascape, where the quantifiable forces of a design project are visually represented and these representations can be used for the conception or development of a design, (iv) Metamorphosis, which uses an interpolated state of a morphed form in the conception of a design, (v) Genetics, where genetic algorithms are used

to simulate the concept of evolution with computer logic to produce prototypical forms in response to their simulated environment and, (vi) Performative, which uses the computer's capabilities for the simulation of a targeted performance to generate building forms in response to its simulated environment (Kolarevic, 2003).

Flanagan (2005) presented four generative fundamentals in digital design, (i) 'Generative Geometry', through which a single structural component can be substituted or changed, thus systematically and simultaneously affecting the overall structure and space in an 'organic response', (ii) 'Generative Imagery, which is a translation of a two-dimensional representation of structure and space by repeating blocks which have scaled variants, (iii) 'Generative Manufacture', which employs the 'lowest common design dominator' (LCDD) with generative rules of construction and, (iv) 'Generative Space-time', which adopts the time concept as an added design dimension by incorporating multi-sensory design variables (Flanagan, 2005).

The three approaches put forward by Rowe, Kolarevic and Flanagan and discussed above all share some common elements. Which of these fundamentals is used is based on the designer's preconceptions, but any one of them could be used to formalise the design concept.

The pre-design phases of briefing and analysis are essential in the design process in addition to the final product. The outcomes of these phases extend throughout the design process. The designer's guiding principles influence the prioritisation of the design constraints according to the project's goals and objectives. The subjectivity in the prioritisation of the design constraints influences the capture of the design concept. The design concept is usually chosen based again on the designer's guiding principles (Frazer, 2002). Abstracting the captured design concept, the designer's primary generator influences the conception of the design schema in the reflection to the third type of constraint in Lawson's (2006) observations,

which is derived from the designer's stance and is thus a 'self-imposed' constraint (Janssen, 2004).

4.5.3 Designer's schema

In a general sense, a schema is a plan or a model. Examined in relation to human psychology, the simplest way to describe the schema is as a representation of the way an individual organises his world-view, which includes past experiences and learning in addition to pre-conceived ideas. The schema is often seen as the organisation of knowledge — past, learnt and context specific — which is used in the interpretation and processing of information. As Bartlett puts it, the 'schema refers to an active organisation of past reactions, or of past experiences, which must always be supposed to be operating in any well-adapted organic response.' (Bartlett, 1995, p. 201). Through his studies on remembering, Bartlett found that if an individual is unfamiliar with a particular subject, their memory (and thus their understanding) of it would not be as meaningful as that of someone who has developed an appropriate, context-related schema. Based on Bartlett's work on memory and the cognitive process, Lawson describes the schema as the structure of past experiences that individuals use in arranging and understanding or interpreting future events (Lawson, 2006).

In the context of design, the design schema, as described by Frazer, is a 'highly personalised but generic methodology' employed by most designers (Frazer, 2002, p. 259). Because it is an adaptable design model inserted into the design process of each designer, the schema characterises the designer's style as an abstract conception of the common features of their designs (Frazer, 2002) (Janssen et al., 2002). Janssen notes that the schema is commonly seen as being developed throughout the designer's practicing career, in reaction to 'niche' environments, which are 'a range of design environments that are similar to one another' (2004, p. 163).

Each project or design problem faced by an architect is unique in one way or another. Even taking the similarities of some aspects of different projects

into consideration, each project is found to be distinct in the problems it raises. Thus, applying the design schema, which is developed out of past experience of similar 'niche' environments, to every similar design problem has the potential of creating contradictions and conflicts with the unique aspects of an individual project. Although the outcome solution may well be successful, it runs the risk of being tedious or lacking in creativity. To overcome this, designers, armed with their schemas, tackle each design problem with a specially conceived 'design schema'. A distinction can thus be made between what we can call the 'designer's schema', which we will understand as the more general and all encompassing 'personal style' or 'personalised methodology' of the designer (albeit in relation to a 'niche' environment), and the 'design schema', which is project specific and is developed to respond specifically to the 'targeted' environment of the unique design problem at hand.

4.6 Design schema

The 'design schema' is introduced here as a project-specific design model, framework or process, which gives free reign to the dynamism and fertility of the designer's creativity. The design schema is developed with, but it is not limited to, the designer's schema. When faced with the unique design problems of a specific project, a designer draws on his preconceptions to interpret and analyse the new information associated with the project. Whether this new project-specific information involves issues the designer has not dealt with before or whether the project issues and constraints are familiar but grouped in an unfamiliar manner, the unique design problems together with the designer's guiding principles, primary generators and personal schema will produce a distinct, process-oriented design-specific schema. In other words, it is the interaction of the designer's preconceptions, guiding principles, primary generators and personal schema, with the project specific constraints and problems that results in the 'design schema'.

The word 'tectonics' originates from the Greek word tekton, which involves all sorts of art and craft (Frampton, 1995). Tekton also encompassing

poesies, which refers to the poetic expression of the maker (Heidegger, 1971). The relationship between the concepts of poesies, which is understood to be the artistic aspect of design, and the concept of tekne, which is the technical realisation of the design, are determined and executed in the architecture. In *'Structure, Construction and Tectonics'*, Eduard Sekler defines tectonics as 'a certain expressivity arising from the static resistance of constructional form in such a way that the resultant expression could not be accounted for in terms of structure and construction alone' (Frampton, 1995, p. 19). Karl Botticher, who coined the term tectonics in his work, *The Tectonic of the Hellenes* (Hübsch, 1992), made the 'seminal contribution of distinguishing between the Kernform [the core form] and the Kunstform [the art form]; between the core form of the timber rafters in a Greek temple and the artistic representation of the same elements as petrified beam ends in the triglyphs and metopes of the classical entablature' (Frampton, 1995, p. 4). Tectonics is an integrated design realization of aesthetic expression and technical practice. Exploring positions on tectonics from aesthetic analysis to efficient physical structure, Frampton states that, 'everything turns as much on exactly how something is realized as on an overt manifestation of its form. This is not to deny spatial ingenuity but rather to heighten its character through its precise realization' (Frampton, 1995, p. 26). In this sense, tectonics realises poetic resolution in collaboration with the technical resolution.

It is not only the physical reality of a building that is encompassed by the cognitive technical practice. The functionality of the use and purpose are also within the realm of 'cognitive technical practice'. For a design to be considered to be a well-planned and conceived tectonic design, it must incorporate the search for form and art within the integration of the technology and the technical knowledge. In tackling a design from a technical perspective, the technical process itself involves artistic expression. Thus, we might say that poesies is achieved through the process of searching for the right tekne solution for the unique needs of a project. More simply put, even during the process of creating a functional,

technologically sound and economical solution, poetic expression can be achieved. Incorporating the tectonic approach from the beginning can produce integrity in the building.

Picon (Cited by Lim (2009)) argues that the designer's classical perception of the world has undergone a change from the laws of order and proportion to a new version that is characterised by changes and movement. Thus, designers have changed their conception of the basis of the built environment from the classical view, based on the geometrical realization that defines beauty in reference to an object, to a more scientific view, based on mathematics and algorithms, that adopts the new definition of beauty based on a dynamic processes of constructive patterns (Lim, 2009). This introduces a design approach that falls in-line with the tectonic realisation. A design approach which places higher emphasis, recognition and value on the aesthetics of the process and organisation of the designed product. Process-oriented design is the new form of technological thought that overlooks the old object-oriented design process.

Throughout the conventional design method, the architect designs a set of tools that help produce a design solution. In this sense, the design concept, primary generators and the design schema are design tools. Design-tool making, whether conventional or digital, thus involves the capture of a design concept, which is then interpreted and abstracted into a design schema. The subjectivity in the prioritisation of the design constraints influences the capture of the design concept based on the designer's preconceptions. The guiding principles of the designer's preconceptions influence the designer's prioritisation of the design constraints according to his interpretation of the design goals and objectives. In response, the designer produces a design concept based on his primary generators, which is then elaborated into an executable design schema.

The design schema presented here is intended to respond to the uniqueness of the manifold design problems faced by different designers. As the design schema is the result of the interaction of the designer's

guiding principles, primary generators and the designer's schema with a specific design problem, it is dynamic and will clearly reflect aspects of the designer's creativity. With regard to computer-aided architectural design, the challenge is thus to find a way to incorporate the flexibility and creativity inherent in the design schema into a language that the computer can understand. The proposed design schema intends to facilitate this process. The anticipated design system acknowledges and respects the need for the design schema to be unique in each design problem and to each designer to retain the creativity of the designer in the design process.

When the design schema as a process-oriented design is employed in finding a solution to a design problem, it helps the designer to maintain his creative expression as a central part of the design process. The design schema accommodates the continuance of human creativity in the invention of new design tools. The proposed design system being researched takes advantage of the genetic evolutionary algorithm as the main engine for the generation of design solutions. The use of genetic evolutionary algorithms also aids in the optimisation of the successful evolved models in response to the simulated environments interpreted out of both the external and internal constraints of the design problem.

The design method seeks to maintain the flexibility and uniqueness of the design schema by using definite clause grammars (DCGs), space layout planning with graph theory, regulating elements and form generation with modular structural component elements. It is anticipated that this combination of processes will allow the designer to express his unique creative character in the development and subsequent translation of the design schema. This also means that the design schema does not have to be limited or restricted, thus allowing a vast variety of unique designs schemas to be conceived.

4.6.1 Definite clause grammars

Definite clause grammars are the grammars used in logic programming. As their name suggests, they define clauses and can thus been seen as an

extension of context free grammars or phrase structure grammar. The history of definite clause grammars is closely related to the history of Prolog, 'Programming in logic'. Prolog's grammar rules provide a convenient notation for expressing definite clause grammars.

Prolog, developed in 1972 by Alain Colmerauer and his colleagues at the University of Aix-Marseille in France (Frampton, 1995), is a programming 'language that is used for solving problems that involve objects and the relationships between objects' (Clocksin & Mellish, 2003, p. 2). Since its introduction, Prolog has been widely accepted and adopted for use with knowledge-based systems and in artificial intelligence applications, particularly automated reasoning systems (Frampton, 1995).

Prolog, based on formal logic, is a versatile language that can be used to 'implement all kinds of algorithms, not just those for which it was specially designed' (Frampton, 1995, p. 1). While Prolog can be used to solve problems from an 'object-oriented' perspective, it is not, an object-oriented language, because it can be applied in other ways. It is considered to be a 'more versatile solution to the problem that object orientation was designed to solve'(Frampton, 1995, p. 2). Prolog is a declarative programming language, which means that the problem can be directly expressed in the form of facts and rules (which reflect the properties and relations of the problem, respectively) (Frampton, 1995). Using imperative languages such as C or Java, on the other hand, involves specifying how to achieve a certain goal in a certain situation. When implementing a solution to a problem in Prolog, the situation is specified in terms of 'rules' and 'facts' and the goal is laid out in the form of a 'query', which then allows Prolog to solve the problem.

According to Covington, Nute & Vellino (1995), Prolog can be seen, above all, as a language that enables the representation of knowledge. They point out that Prolog is essentially different from conventional databases because, in Prolog, whether the knowledge is inferred or deduced, it is given the same status as explicitly-stored information in the knowledge

base. In other words, while Prolog is capable of distinguishing whether or not a query is successful and with what variable instantiations the solution is associated, it does not distinguish between whether the solution was found directly from the knowledge base or whether it was computed by inference. Because Prolog deciphers clauses as procedure definitions, it requires both declarative semantics and procedural semantics: declarative to represent knowledge, and procedural to be able to prescribe the computational actions. Prolog is, however, 'not perfectly declarative; the programmer must keep some procedural matters in mind' (Frampton, 1995, p. 31). For example, it is possible that a declarative knowledge base could result in a loop of one fact following on from another. It is also possible that various knowledge bases, while all being correct declaratively, may vary greatly in computational efficiency. In addition, to make it possible to use Prolog to write programs that are interactive with the user, as opposed to just writing knowledge bases that are able to answer queries, a more procedural approach is necessary.

Because Prolog is a 'declarative' programming language, it calls for problems to be stated in terms of facts (objects) and rules or conditions (the relationships between objects) that a solution must satisfy. 'The computer can figure out for itself how to deduce the solution from the facts given' (Frampton, 1995, p. 1). Using Prolog programming, the programmer deals with formal relationships and objects, 'asking which relationships are 'true' about the desired solution, thus Prolog is seen as a 'descriptive' language that will allow the designer to 'prescribe' his unique point of view of a problem. The facts and rules of a design problem that are declared by the designer are based on the designer's interpretation of the design environment (which includes the designer's preconceptions, guiding principles, primary generators and the specific design problem). The designer is thus able to inject an element of his creative vision into developing the design schema.

The designer prescribes the building's configurations using the Prolog language to generate a syntactic parse-tree representation. A parse-tree or

concrete syntax tree is a representation of the syntactic structure of a string, ordered and rooted according to a formal grammar. The interior nodes of a parse-tree are labelled non-terminals of the grammar, and the leaf nodes are labelled the terminals of the grammar. The language generated by the grammar consists of all the strings that are classified as grammatical. As a result, different grammatical strings of the building configurations could be produced to govern the Synthesis of the building spaces and components.

4.6.2 Dual graph

At the time when genetic algorithms were first proposed, it was suggested that they should be encoded using binary strings (Goldberg, 1989). The representation of the solutions to some problems (for example, spatial configuration), as linear-string chromosomes can be problematic, however. By the 1990s, various other kinds of evolutionary algorithms, using different forms of encoding, were being discussed and studied. Since the 1990's, Michaelwicz's floating-point presentation in chromosomes, permutation and the matrices presentation of graphs (Michalewicz, 1996) in addition to Koza's tree encoding (Koza, 1994), to name just a few, have, according to Wong and Chan, been successfully developed and used in different applications to solve a variety of problems (Wong & Chan, 2009). Wong and Chan present vehicle routing, the travelling salesman, linear programming, the knapsack problem and multi-object optimisation as examples of the types of problems that can be tackled using genetic algorithms. The ability to use graphs to represent the solutions to problems, such as space layout planning, clearly greatly enhances the use of genetic algorithms.

Dodd suggests that to make use of directed or undirected graphs with genetic algorithms, adjacency matrices can be used. An adjacency matrix can be produced for the space layout problem at hand; this adjacency matrix can subsequently be encoded into linear chromosomes that can then be fed into a genetic algorithm application (Dodd, 1990). Dodd's own method of encoding a graph involves linking the rows of an adjacency

matrix into a linear string. There is, nevertheless, a limiting factor involved, which is for the chromosome to be decoded back into a graph, the length of the chromosome has to be equal to the square number of the dimension of the adjacency matrix (Dodd, 1990). The chromosomes can thus be very long if the graph is large. Wong and Chan point out that not only does Dodd's method not allow for the evaluation of the chromosomes of varying lengths, but also that special encoding and decoding and repair mechanisms are required before and after the evolution process (Wong & Chan, 2009).

Dodd's method of using an adjacency matrix is not the only way to encode a graph for a genetic algorithm. Various indirect methods of encoding have also been developed. In the 1960's, Lindenmayer suggested that encoding could be carried out on a grammar-based construction program in the genome (Lindenmayer, 1968). By the 1990's, Kitano was suggesting that encoding could be achieved through the evolution of graph-generating rules (Kitano, 1990). Other indirect methods of encoding graphs involve instructions from genotypes being read in the form of either chromosomes (Arbib, 2003) or a tree (Gruau, 1993). In the case of Arbib's use of chromosomes, a genetic algorithm is used at the genotype level to perform evolution. Gruau's use of a tree, on the other hand, uses genetic programming. The problem, as Wong and Chan point out, is that the search looses efficiency through the 'indirect evolution of the genotype or evolving graph generating rules instead of the evolving [of] the graphs themselves or their direct representations' (Wong & Chan, 2009). In terms of the proposed design system, to avoid any additional or special encoding and decoding, it is proposed that graphs are encoded using adjacency matrices. This will also allow for graphs to be generated directly without taking into account any special operators of crossover or mutation.

All physical design problems, whether component packing (Yin & Cagan, 2000), route path planning (Shaikh & Kang, 1997), VLSI (Tang & Yao, 2007) or architectural layout design (Levin, 1964), require spatial configurations. A problem emerges when automating spatial configurations because it is

necessary to place the interrelated objects that the design requires into locations that are feasible, while also ensuring that the quality of the design is maximised. In the case of architectural layout design, matters are made even more complicated because the components are rooms, walls and such like and, as such, do not have a set or pre-defined dimension, meaning that all components need to be resizable. In addition, architectural layout design takes into consideration not only the engineering requirements such as the cost and performance, but also the aesthetics and usability that are almost impossible to define and describe in the formal language of a computer.

The use of dual graphs within the proposed design method — whereby the spatial organisations are graphically represented according to the adjacency constraints between the spaces and in line with the regulating element — formalizes the designer's vision of the building's spatial configuration in an automatable representation for the genetic algorithm system. Each graph is translated into orthogonal rectangular duals and encoded into the genetic algorithm system with an expected performance criteria set to govern the evolution. To explore the larger space of graph topologies, the adjacency matrices and reproduction operators of the graphs can be topologically configured and encoded for use by the genetic algorithm system.

4.6.2.1 Rooms space layout planning

In a study of functional requirements in spatial design based on the field of Ambient Intelligence, Bhatt et al. (2009) define architectural space based on conceptual, qualitative and quantitative 'Ontological Modules'. The conceptual ontological module takes into consideration only the most general aspects of the architectural entities. From a conceptual stand point, the functional requirements of an architectural space are visualised and conceptualised without the contextual consideration and are seen only in relation to their base and essential properties. Architectural entities are grouped according to their functional characteristics, attributes, dependencies and so on. The qualitative ontological module incorporates the qualitative spatial data of architectural entities. Architectural entities

are viewed through their spatial characteristics that are influenced by their region and relation to other spaces. The quantitative ontological module takes into account information concerning the architectural entity on a metrical and geometric basis. This covers the polygon-based characteristics in the floor plan of the entity. The quantitative ontological module is related to the Industry Foundation Classes (IFC) that are an architectural industry standard for data representation and interchange.

Bhatt et al. (Bhatt et al., 2009; Bhatt, Hois, Kutz & Dylla, 2010) identify three key categories of spatial artefacts, object space, operational space and functional space. The object space of a basic architectural entity is associated with the space incorporated within the physical aspect of the entity itself. It is simple to define objects that are fixed, unchanging, and cannot be deformed or reconfigured. However, an entity that is dynamic, can be deformed and reconfigured has spatial extensions that are dependent on the particular state(s) it is in. Bhatt et al. offer the example of the angle of an opening of a door or window. An object or entity's operational space deals with the region of space required by that object to fully perform its fundamental function or purpose. The example given by Bhatt et al. (2009) is again that of a door that opens in only one direction. The operational space is the space required to allow the door to be freely and easily fully opened, fully closed and the space required for the door to move between these two states. The functional space of an entity is the surrounding region of space required for an agent to be able to physically interact or manipulate with the object in question. When agent interaction with an object is minimal, the functional space can be approximately the same as the object's convex hull, but it does not necessarily have to be similar to the object's convex hull.

The whole building is comprised of a number of rooms, each of which is, in itself, comprised of a collection of building components, fixtures and furniture or, in some cases, a space for some type of activity that the room is intended to host. The spatial artefacts of each entity (object space, operational space and functional space) form an over-all space (a room)

that is usually rectangular. The room is, in fact, a packing of the object, operational and functional spaces. Similar to the space layout planning of the rooms to generate the building floor plan, but stepping down a level, each room could be generated using the same technique. The result would be a number of variants generated for each room that would be functionally acceptable.

Rather than dealing with fixed dimensions and an area, or even a range of minimum and maximum widths and lengths assigned for each room, the idea is to have a more flexible building-layout-generation packing process. This flexibility would allow for the selection of the most suitable room based on the actual functional requirements of the room, whilst taking advantage of the inherent freedom of selection.

A dual graph for each room could be generated according to the spatial organisation of its constituent spaces. This spatial organisation is governed by the adjacency constraints of the room's required entities that are, in turn, governed by the functional role that they play in facilitating the expected activity that the room is being designed to host. As a result, different room variants can be generated with each representing a specific set-up of relationships between the room's entities. Alternative room variants can also be generated if and when an overlapping between the operational spaces or the functional spaces is permitted.

The space layout planning process is thus the starting point for the composition of the building from the lowest level up. The genetic algorithm is expected to facilitate the space layout planning of the whole building on both levels, the room and the building level.

4.6.3 Regulating lines and regulating elements

For every 'massing configuration', which Akin describes as 'the primary sub-set of the early stages of build form creation' (Akin & Moustapha, 2004, p. 31), there is an underlying structure. In architecture it is the norm to use

regulating lines to depict the structural relationships of massing elements. According to Le Corbusier (2008, p. 137):

> A regulating line is a guarantee against arbitrariness… the regulating line is a satisfaction of a spiritual order that leads to a search for ingenious relationships and for harmonious relationships… the regulating line brings forth the sensory mathematics that produces a beneficent perception of order. The choice of regulating line fixes the fundamental geometry of work. The choice of regulating line is one of the decisive moments of inspiration, it is one of the crucial operations of architecture.

In his article, '*Durand and the Science of Architecture*', Madrazo (1994) investigates Durand's attempt to develop a theoretical architectural system. Durand's method was illustrated graphically in six stages where the first stage is setting the main axes of the composition. In the second stage, a new grid of secondary axes, complementing the primary axes, is set. The approaches of both Le Corbusier and Durand demonstrate the role that the regulating lines have played in the development of architectural theory.

Akin and Mustapha (2004) conducted an empirical study on 'architectural massing', which occurs during the early stages of design activity. They extended the concept of 'regulating lines' to 'regulating elements', which include points, planes and volumes. Site planning and design, space layout planning and elevation design influence the massing decisions to a great extent. Regulating elements such as axes of symmetry, centre of rotation, alignment axes, diagonal proportion lines, points of intersection and bounding lines are usually used as a mechanism in structuring massing strategies. It was observed during their protocol studies that these strategies are implemented in a regulated design process to administer the part-whole relationships, design hierarchy, scaffolding the design process, topology-geometry relationship, structuring sub-problems and the structuring of the design parameters (Akin & Moustapha, 2004).

These strategies, of regulating the elements, managed and directed by the regulating design process, are seen here as foundations for space layout

planning using graph theory. They also provide a base for the overall configurations of the design solution that is governed by the Prolog description of the building. The regulating elements provide the guidelines for the translation of the parse trees generated out of the Prolog grammar in a non-arbitrary spatial configuration. The regulating elements introduce the basis for the expression of the entire structure of the architectural configuration in three dimensions by relating the massing configuration and organising the elements of massing configurations. They also form the spatial sub-division within masses and relate massing elements to their constituents.

4.6.4 Lowest Common Design Denominators (LCDD)

The lowest common denominator or least common denominator (LCD), in the field of mathematics, is the lowest common multiple of the denominators of a set of common or vulgar fractions (*The American Heritage Science Dictionary*, 2005). The LCD is the smallest positive integer that is a multiple of the denominators. The 'lowest common denominator' can also be used in everyday English to meaning the most simple or basic shared interest or characteristic among a group or collection of people (Hirsch, Kett & Trefil, 2002).

In the field of design, the Lowest Common Design Denominator (LCDD) is a shared block divisor where its multiples constitute the whole (Flanagan, 2005). Lowest Common Design Denominators (LCDDs) is a reduction method that attempts to decompose a formation to the smallest common block. The level of the decomposition of a design configuration can, potentially, go to the most basic geometry, such as a rectangle, a triangle or even the quarter of a circle. A modular approach to design is, in fact, a lower LCDD level decomposition of the overall formation, because it can be formulated from multiple geometries. The composition of a formation is achieved by a repetition of a number of LCDDs in a certain order of arrangement. Identical LCDDs and/or non-identical LCDDs that are manipulated by different transformational processes, such as scale, stretch,

rotate, mirror, and so on, can develop, through the process of multiplication, into a structure.

The idea here is to generate the two-dimensional and three-dimensional form modules from a selected LCDD. The LCDD is a geometrical seed designed with a transformation rule(s) to govern its evolution. The two-dimensional LCDD module is used for the space layout planning, where the parameterised structural module is used for the massing of the design solution. A variable set of parametric rules and mechanisms, which allow for a diversity of unique building components, defined the modules.

4.6.4.1 Two- and Three- Dimensional LCDD

According to Agkathidis (2009), Vitruvius, in his analysis of Greek temples, brought modularity in architecture to light when he introduced a measured module unit that defines the relationships between all components and the rest of the building as a whole. Repetitive patterns of geometric blocks are an approach that has been used to create artefacts in many disciplines, such as art, textiles and in ornamental design, for many centuries (Buck, 1915; Day, 1999; Kaplan, 2000; Kaplan, 2002). This approach has also been applied in architectural layout planning (Steadman, 1983), form generation, building facades (Flanagan, 2005). The modular system was applied in a two-dimensional pattern, such as the traditional tatami mat, as a predecessor to modular planning (Kuroishi, 2009).

Today's rapid advancements in the fields of information systems transform the architectural, engineering and construction industries by incorporating a new perception of modularity. Algorithmic and parametric tools in the applications developed for architectural design and manufacturing technologies revolutionised the production of the construction elements. Modularity is no longer a static term, but more a new notion for design realisation that allows for economic and efficient building design and construction. The new module is defined with a variable set of parametric rules and mechanisms allowing for a diversity of unique building components, all of which are inherited from the same associative module

(Agkathidis, 2009). When the designer defines or declares the parameters of a design, he can produce a model that includes variable geometrical definitions and behaviours. This parametric design method enables the designer to define the relationship between the various design components of a parametrically defined model and, in addition, a set of constraints or rules can be set that govern the variables which, in turn, means that the model can be manipulated through the design process without any loss of control over the design itself (Shah & Mäntylä, 1995). The outcome can achieve the desired efficiency and cost limitations through a parameterised module, which combines technology, complex geometry and ornamental aesthetics into one particular entity (Agkathidis, 2009).

In approaching the generation of the design LCDD, several methods can be used. For instance, the designer may choose a two-dimensional LCDD module that is either generated based on a specific reference originating from the design context, such as the geometric properties of the site, or some kind of functional or spatial requirement imposed as a design constraint. The designer also may create a two-dimensional LCDD module that is conceived purely from his own preconceptions and creative impulses. In all of these cases, the chosen two-dimensional LCDD module can be considered to be the basic seed for generating the overall grid system with respect to the regulating lines and elements of the site, and also to generate the modular structural component. The assignment of the transformational rules for the LCDD to evolve into patterns and composition is another means for the designer to express his creativity, reflecting his preconceptions throughout the design process.

The two-dimensional LCDD, with its transformational rule(s), governs the evolution of rooms in addition to the overall building layout. the evolutionary genetic algorithms can control this process of morphogenesis to generate room variants using the layout planning method. Carefully assigned fitness functions can be used as a basis for the selection of the successfully generated candidates that meet with both the design requirements and the designer's creativity. The design process would thus

be allowed to unfold, based on the spatial functional requirement data, rather than producing a design that would then have to be analysed and optimised by the designer after the fact.

The two-dimensional LCDD module, with specially assigned transformational rules, could evolve into a three-dimensional parameterised structural component through the addition of a height parameter. This would mean that the overall design solution would be consistent, because it would be generated out of a single, original two-dimensional LCDD.

4.6.4.2 Parameterised modular structural component

The flying buttress (Adams, 1991) and the Muqarnas (Castéra, Peuriot & Ploquin, 1999; Frishman, Khan & Al-Asad, 1994) are examples of the early use of the three-dimensional modular structure. During the industrial era, modular construction was developed to accomplish easier production, quicker transportation and cheaper assembly. The identical modular units of the balloon frame and, later, its optimised version, the general panel system, achieved the goals that they were developed for and dominated architecture for several decades (Agkathidis, 2009).

Flanagan (2005) explores the applications of the modular principles of conventional architecture in computer aided design. Notwithstanding the limitations of the CAD technology available at the time, Flanagan conducted a series of experiments from 1995 to 2002. As illustrated in his article, he demonstrates an alternate approach to modular design by incorporating the advantages of traditionally-manufactured modules, while benefiting from the computer's capabilities at the same time, which allows for greater design flexibility. Flanagan argues that the rationalisation of the design process in the prevalent application of CAD in architecture indicates the declining impact of form generation in building design, while also indicating a greater reliance on information systems in building functions. Yet, this new trend requires the architect to negotiate imbalances in form,

function and technology, because they prove to be the essential forces in the generation of architectural expression (Flanagan, 2005).

Over the last two decades, a large number of encouraging advances have been made in the search (in both academic research and in practice) for design software that can use declared parameters to connect the implicit geometrical relations (Shah & Mäntylä, 1995). Advances in automation capabilities, in addition to the vastly increased computational speeds now possible, mean that results can be generated much more quickly than ever before. As Holzer, Hough and Burry (2007) point out, the software that is now available to engineers can be used for complex and time-consuming activities, such as calculus and code-checking, and can produce results quickly and accurately during the early stages of the design process. The speed at which the results can be obtained allows the decisions on the structural design, for example, to be supported by the results obtained from the optimisation carried out by the structural engineers. Structural design and structural engineering could, in fact, be done in parallel, with sound structural engineering information being used as an informed base for the structural design.

Although much of the software currently available to architects is limited to geometry generation in drafting, (and, to a more limited extent, in design), and the software available to structural engineers for analysis and optimisation is, for the most part, geared towards specific structural tasks. Architectural design could gain a great deal from software that could interface between and across all disciplines. As Holzer points out, 'the upgrade of the role of the computer from being a design assistant to being a design collaborator is possible through a tight link between associative geometry, structural performance evaluation and structural optimisation' (Holzer et al., 2007, p. 631). Unsurprisingly the potential of linking parametric design with engineering analysis and optimisation processes has been the subject of much research. Running parametric design software in parallel with engineering optimisation and analysis software could allow the design process to unfold, based on the engineering data, rather than

producing a design that would then have to be analysed and optimised by engineers after the fact. As Holzer explains, 'by linking parametric design to structural analysis and optimisation, architects and structural engineers can explore design in the conceptual design phase through informed geometry alterations' (Holzer et al., 2007, p. 640).

The parameterised modular structural component transforms the generated spatial configurations into a three-dimensional model following the regulating elements as a basis for the expression of the entire structure of the architectural configuration. This is achieved by relating the massing configuration, organising the elements of the massing configurations, forming the spatial sub-division within the masses and relating the massing elements with their constituents.

4.7 Encoding

Architects 'deal with approximate knowledge and require an open logic system' while the computer 'operates on a closed logical system' (Flanagan, 2005). Many of the constraints that arise during the design process are factual (environmental factors or site context) and can thus be entered in a way that the computer systems can understand. Other constraints, however, are less absolute and more ambiguous, making them much more difficult to encode into a logic-based computer system. A self-imposed design schema to generate design ideas is also based on subjective, approximate factors. All these constraints need to be represented and encoded in a way that the computer system can understand.

The design rules and representations for genetic algorithm systems can be encoded in one of three ways (Janssen, 2004). The highly generic approach applies standard rules and representations using binary strings that do not rely on any domain or task-specific knowledge. The performance of an evolutionary algorithm in generating designs using a highly generic approach, based on broad generalisations, is not necessarily highly efficient. However, the lack of performance is compensated by re-usability. The domain-specific approach incorporates the domain-specific knowledge

in the rules and representations to improve the performance of the system. Unfortunately, there is a corresponding loss in re-usability. Finally, the task-specific approach can be used when the type of task in the domain is complex and a high level of performance is required. The task-specific approach is at the opposite end of the spectrum to the generic approach. While it offers the highest levels of performance, re-usability is very low.

In the domain of building design, when the task-specific rules and representations of a specific designer are encoded, the system will reflect the designer's preconceptions and style. Therefore, the domain-specific approach is found to evolve more generic designs and it is also more flexible for re-use by other designers. In addition, the domain-specific approach evolves designs with high variability. Even so, the domain-specific approach produces designs with a poor efficiency. The task-specific approach on the other hand, compensates the re-usability and style issues by evolving more unpredictable, novel and challenging designs with high efficiency (Janssen, 2004).

When using a computer-aided space layout program, the quality of the solution to the layout problem hinges upon the information entered into the program by the architect. The decision regarding which form the initial input into the program should take is identified by Ruch (1978) as one of the biggest problems faced when using a space layout program. This is because the information the architect has at this early stage, which the input decisions necessarily have to be based on, can often be incomplete or, in some instances, over-specified. Placing fewer constraints on the definition of the initial input has the positive aspect of allowing the program to produce a large array of alternative layouts that comply with the initial input requirements. Rosenman argues, that the less that is initially known and entered into the program about the requirements and the form, the more creative the design solution will be (Rosenman, 1997). Ruch (1978) points out, however, that there is a negative side to this, in that there may actually be too many alternative layouts produced, and even with the help of a computer, these may be too hard to enumerate.

In an effort to get over the hurdle of incomplete information and definition, space layout programs usually require very specific data to be entered. For instance, the overall shape of the building and the exact shape and size of the spaces that make up the building may need to be predefined. While the more precise data input requirements can aid in the program, producing a set of alternatives which are more targeted and thus more manageable, the requirements for which are much more specific than those required by the typical architectural program. In addition, there is the danger that a definition that is over-specified could, potentially, contain contradictory information. If that were the case, then the program would not be able to generate a successful solution to the layout problem. As observed by Rush (1978), many programs attempt to get around this problem of over-specification by using a system where the designer can set a range of priorities for the requirements. In a case of an over specified problem, the computer will deal with the requirements assigned the highest priority (Ruch, 1978). Ruch makes it clear though, that because most of the information needed to solve these problems is unusual in the architectural program, 'these approaches result in the input becoming an arbitrary limitation on the possible layout results' (Ruch, 1978).

Clearly a balance is needed here. When predefining part of the solution, the expression of the designer's creativity is constrained, because the system only searches for solutions from within the space of the constituted building based on the predetermined solution. Allowing the genetic algorithm to generate a variety of parameterised spaces according to the predefined set of rules and assigned fitness functions, rather than the solution itself, enables the system to choose from a variety of different shapes and sizes of each space to generate a variety of layout plan configurations. In addition, the weighted adjacency matrix together with the prioritisation of the design constraints offers a far sounder direction to take when seeking to resolve the problem of the over-specification of the design requirements.

The adjacency data in the graph may be defined as either true or false, which does not allow priorities to be defined. Tarjan's conception of depth-first research (cited by Wu et al. (2004)) facilitates the search of all possible paths between two nodes of a weighted graph, according to a weighted adjacency matrix. The designer sets the adjacency rules and requirements for the design based on the spatial functional relations between the building spaces. The use of the weighted adjacency matrix supports the generation of alternatives and therefore could generate different graphs.

All of the project's identified constraints are encoded according to the relative priority assigned to them by the designer. The system will act in favour of the constraint that is assigned the highest priority. For example, if the building orientation is determined to be more important than the environment, the environmental solution will be applied to the best orientation solution. On the other hand, if the environmental constraint is set with a priority higher than that of the orientation, the best environmental setting determines the orientation. Thus, each set of priorities produces a different generation of solutions with a set of different characteristics.

The prioritisation of the project's constraints and the subsequent design solutions synthesised in accordance with the simulation of the project environment during the design generation is expected to resolve most of the major conflicts in the evaluation and optimisation phases. The system produces a responsive design solution, optimised and developed through its creation process in simulated and animated environments, thus being responsive to its environment, performance requirements and to the designer's creativity (through the design schema).

4.8 Generative evolutionary architecture

Nature, ever changing and evolving, serves as a rich source of inspiration for designers and researchers attempting to optimise the performance of their designs. This is not a new concept, John Frazer used the phrase 'Evolutionary Architecture' (Frazer, 1995; Frazer, 1974; Frazer & Connor,

1979), referring to the investigation of, and search for, form-generating processes in architecture, based on the notion of morphogenesis in the natural world. Frazer's conceptualisation of architecture goes beyond the idea of shapes and forms to encompass a set of rules that generate spaces and forms. The built environment is thus produced through a series of evolutionary steps based on the selection of solutions that best respond to the fitness function criteria.

Genetic algorithms are used to simulate the concept of evolution with computer logic. John Holland first presented genetic algorithms in the 1960s in his investigation of the process of natural selection systems (Holland, 1992). As described by Holland, the structure of genetics consists of a population of chromosomes that represent possible solutions to a problem. Genetic operators, such as crossover and mutation between two or more genotypes, produce a new generation of phenotypes.

Crossover takes place when parts of the genomes (the encoding) of two selected individuals, or 'parents', are exchanged to form two new individuals or 'offspring' or 'children'. At the most basic level, after a crossover point is randomly selected, 'substrings' can be exchanged. Crossover is a very useful operator that can be used to search for new and potentially promising aspects of a search space. The mutation operator, on the other hand, can be processed by 'flipping bits at random, with some small probability' (Wong & Chan, 2009, p. 649). The operation of mutation, which randomly samples new points in the search space, is used in an attempt to stop convergence to a local optima occurring prematurely. Because genetic algorithms work through many iterations at random, there may never be convergence and thus the termination of the algorithm may have to be specified by the designer. This termination condition could be anything from setting a maximum number of generations to setting an acceptable level of fitness (Wong & Chan, 2009).

The genetic make-up of the next generation is dependent on the process of selection that decides which part of the population will pass to the next

generation. Every solution in the population is evaluated and selection is made on the basis of fitness. Genetic algorithms deal with a huge number of possible solutions as opposed to just one solution at a time; therefore, it is much more likely that an acceptable solution will be found (Kalay, 2004, p. 283).

Although the genetic algorithm incorporates a generative capability, it requires an initial generation to start the evolution process to produce the next generation. The more diverse the initial generation, the more varied is the generated population. Initiating the evolution process with a wide selection of parents of different characteristics enriches the propagated generation. The parametric generative design algorithm, with its capability of generating design solutions based on transformational rules, provides the genetic evolutionary algorithm with an initial generation as a starting point. The genetic evolutionary algorithm can then expand the optimisation search process throughout the solution space of the evolved populations to evaluate and select successful and fitter solutions for the next generation. Thus, we can distinguish two types of operations, namely, the Generation and Formation, and the Evaluation and Optimization.

4.8.1 Generation and Formation

As Ruch (1978) points out, architectural design is, by its very nature, evolutionary. At the beginning of the architectural design process it is very often the case that the architect has very little information about the 'design problem' or the architectural needs of the project. It is during and throughout the design process, when the designer looks for solutions to the design problem, that he identifies the architectural requirements of the building. As the design process unfolds, it is not uncommon for the designer to recognise additional requirements that emerge as a consequence of work that has been done. The architectural design process thus 'tends to be interdependent, with each subsequent decision depending on the partial layout already developed' (Ruch, 1978, p. 152). It is not surprising, then, that space layout programs that rely on all information and decisions being made and entered into the system at the beginning of the layout

development process are extremely limiting to the user, and to the quality of the subsequent designs produced.

The proposed system is designed in the hierarchical levels reflecting the inter-dependence of the design process. Only the information needed to resolve the sub-problem at hand is required at each level of the system. Specific, targeted information would be made available at the level of the system tackling each part of the design problem. The decisions made on the partial solutions generated by each level of the system would subsequently be the input into the next level of the system, together with any additional information needed for tackling the problem at the next level. The proposed system is thus not greatly dissimilar to the traditional design process, in that the layout produced by the system would go through a number of stages of abstract representation. The proposed system has three phases of representation, namely representation through graph theory, which can then be transformed into an integer string representation, and which will ultimately be developed into a schematic plan.

The system generation mechanism uses graph theory for the space layout planning, while it is administered by the DCG as a higher seed governing the general prescription of the building as envisioned by the designer. The regulating lines and elements provide a reference of organisation and the LCDD, together with its transformational rules, generate the two-dimensional module and the three-dimensional structural component that are the lower seed of the building constituents.

4.8.2 Evaluation and Optimization

The proposed design system is targeted at developing a balanced design solution that takes into account the performance requirements of all the disciplines within the design and construction of the building. The multi-disciplinary nature of any design project means that many distinct tasks will need to be tackled and also that many of these tasks will be overlapping. A holistic approach to design is taken, in the expectation that

such an approach will allow for the successful resolution of the multiple tasks involved.

The challenges faced in attempting to manage the conflicts that emerge during building design and construction, and the efforts made to achieve successful methods of collaboration between all the disciplines involved, are not dissimilar to the approaches that have been successfully implemented in the aerospace, marine and automotive industries (Hussaini, Alexandrov, Institute for Computer Applications in Science and Engineering, Langley Research Center & ICASE/NASA Langley Workshop on Multidisciplinary Design, 1997; Yang, Gu, Tho, Choi & Youn, 2002).

Attempts to include optimisation techniques into the building design process have been faced with many challenges. The large number and variety of variables involved in the design process alone mean that the computational demands made by the optimisation techniques are very heavy. In addition, variations of the design may affect the various design objectives in different ways. Based on the notion that genetic algorithms offer a means of dealing with the vast amount and variety of variables, a great deal of research has been carried out on the applications of genetic algorithms in architectural design, structural design, thermal design and heating, ventilation and air-conditioning (HVAC) design. Unfortunately, most of these applications have limited scope, tending to focus on the performance criteria of a specific discipline within the design process. Similarly, although building design simulations have been established for some time now they, too, are developing and evolving primarily within the context of their own disciplines (Yang & Bouchlaghem, 2010).

While recent research, investigating the problems encountered in the integration of multi-domain simulations, has been successful, the solutions found have had some serious limitations. In their attempt to solve the issues involved in building simulation integration, Yang and Bouchlaghem (2010) developed the Pareto genetic algorithm-based collaborative optimisation (PGACO). The PGACO is a framework built to support

interactions between the tasks of multiple disciplines and to coordinate conflicting design objectives. The PGACO adjusts the stochastic values of the interdisciplinary variables in the first level, and then generates new stochastic values of interdisciplinary variables in the next level. This means that the system is designed to operate on, and optimise, a design solution that is initially proposed, where its interdisciplinary variables (shared and coupling) are determined.

4.8.2.1 Multidisciplinary building performance simulations

Building performance simulation tools have developed since the publication of the Solar Energy Research Institute publication *'Design of energy-responsive commercial buildings'* (Ternoey & Solar Energy Research Institute, 1985). There is, however, still a great deal to be done. At the time of the Solar Energy Research Institute's presentation, there was a call for the re-invention of the design process acknowledging the overall building context. Rush (1986) was the first to introduce the notion of integration, as an independent topic, to the foreground, although many other researchers had given integration a voice before him (Bachman, 2003).

Bachman (2003) identified that integration is a part of the architectural design philosophy, where architects do seek appropriate judgement when dealing with the building as a whole, as opposed to seeing the architectural design as a collection of apparatus, integrated between its external and the internal orders. Thus, integration is not a stylistic or a comprehensive design approach; it is, rather, an evolving and progressively more crucial focus of architectural design. On the one hand, architects balance between the physical, visual and performance-related aspects of the different building systems, envelop, structural, mechanical, interior and site. On the other hand, they deal with the integration of artistic notions and the scientific realisation of the building throughout the design process (Bachman, 2003).

The output from the current simulation tools for building performance, which are intended to support building system performance evaluation, are mainly the numeric charts containing enormous quantities of data, which makes them complicated and hard to use and interpret by the designer. The developers of such tools together with the researchers recognise this problem and have been attempting to resolve this issue by finding methods to combine and process these outputs in a way that the designer can use more easily. Decision support systems, database and data-mining are some of the results of those attempts (Souza & Knight, 2007).

Design should not be dominated by thinking based on the notion of physical structure alone. It should be driven by performance and system-based concerns. However, it is fair to say that there is still a need to find the appropriate criteria that are directly related to the design actions to evaluate the performance and therefore effectively relate the design decisions to the simulation results. The use of building performance tools, mainly in the later stages of the design process, to optimise a few design parameters or to support small design decisions, is considered a poor attempt to reflect the concept of the integration in the building design. Methodologies to integrate simulation tools earlier into the design process have been studied in an effort to solve this issue. The most notable methods that have been presented are, incremental levels of complexity, 'playing around with an idea', simple generative forms and genetic algorithms (Souza & Knight, 2007). The methodological problems of setting up the performance evaluation criteria and subsequently relating them to the design actions are independent of the selected simulation tool.

The proposed design schema operates within an integrated dynamic design methodology in which the outputs, performance goals, optimisation and controls are dealt with at the level of the overall building response. Although the proposed design methodology decomposes the design problem in to sub-problems and divides the building design task into different sub-tasks, it does culminate in the assembly of the design solution

through the hierarchical structure of the evolutionary algorithm that deals with the building as a whole.

Buildings are complex artefacts, with energy systems that are characterised by multiple parameters, such as the occupancy level, ventilation rate, degree of insulation, location of thermal capacity, glazing type, extent of HVAC (heating, ventilating and air-conditioning) provision, level of control and type of fuel, to name a few. Each of the multitudes of parameters has a number of states. This, in turn, means that there will be a very large combinations possible and, as a result, one change made to just one parameter state will have a cascading effect on many other dependent parameters. To add to the problem, deciding on one combination over another while balancing between the cost, performance, energy use and environmental impact is a major task. Computer simulations represent a promising solution to this type of complexity (Clarke, 2001).

Clearly, any computer simulation would be the most accurate if the data entered were completely and absolutely accurate. This is not possible when dealing with early design stage and the 'real world' where nature, the environment and human activity are complex, indistinct and often quite irregular. Because the entered information cannot be absolutely accurate, neither can the solution be completely and absolutely accurate. It is impossible to avoid simplifications, assumptions and approximations being used in the process of modelling, and thus the solutions generated will only be as accurate as the entered information. When the level of accuracy required is high, so must be the effort expended in the simulation. As Hong, Chou and Bong point out, the trade-off between the level of accuracy required and the extra effort involved in meeting that level of accuracy must be rationally considered and controlled when building simulation is carried out (Hong, Chou & Bong, 2000).

Throughout the life of every project the evolution of the design is tracked and checked by conducting various assessments at distinct stages of the project. In the earliest stages, when the conceptual design itself is under

consideration, the currently-available simulation tools are not often used for the assessment. It is most often experienced consultants who carry out the assessment based on their own expertise and experience. If computer-aided assessments are to be used at this conceptual design stage, they will need to be based on the representation of expert knowledge rather than on simulation and this poses a significant challenge. Although attempts have been made at developing assessment 'tools' based on artificial intelligence (AI) none have been successful because 'the need for early expert intervention is greatest if the problem is complex and novel' (Augenbroe, 2002).

Despite the problems encountered, computer-based building simulation is commonly applied in the life cycle analysis of a building, including the design, construction, operation, maintenance and management phases. Several popular applications are in use, such as building heating/cooling load calculation, energy performance analysis for design and retrofitting, Building Energy Management and Control System (EMCS) design, complying with building regulations, codes and standards, cost analysis and Computational Fluid Dynamics (CFD) (Hong et al., 2000). Although there are many simulation programs used in the building design, relatively few are available in the public domain. EnergyPlus, one of the more accessible programs available, is a building performance simulation program that combines the best capabilities and features from the Building Loads Analysis and System Thermodynamics (BLAST) developed by the University of Illinois for the U.S.A. Army Construction Engineering Research Laboratories (CERL) and the DOE-2 (Crawley, Lawrie, Pedersen & Winkelmann, 2000).

Although no one program is capable of running every kind of simulation, EnergyPlus attempts to cover as many building and HVAC design options as possible, either directly or by linking to other sub-programs. EnergyPlus has an extended and developed capacity for simulations including integrated simultaneous solutions, sub-hourly, user-definable time steps, ASCII text-based weather, input and output files, a heat balance-based

solution, transient heat conduction, improved ground heat transfer modelling, combined heat and mass transfer, thermal comfort models, anisotropic sky model, advanced fenestration calculations, day lighting controls, loop-based configurable HVAC systems, and atmospheric and pollution calculations. For other simulations that are not a part of the EnergyPlus capabilities, such as thermal loads and/or energy consumption over either the short (a design day) or a long period of time (up to, including, and beyond a year), EnergyPlus links to other popular simulation environments/components. All these simulation capabilities greatly enhance the ability of EnergyPlus for modelling the water and energy flows (such as heating, cooling, lighting, ventilating, and so on) of a building (The University of Illinois & The Ernest Orlando Lawrence Berkeley National Laboratory, 2010).

Because EnergyPlus is an open-source modular program, it also allows the developers to add new features and modules that, in turn, means that links to other programming elements are also greatly facilitated. (Hong et al., 2000). The proposed design method employs the simulation of the project's environments as a tool to aid in the generation of the design solutions, by linking the simulation programs to the Evolutionary Design Systems as fitness functions, to assess each member of the evolved generation. The selection of the survivor would then be based on the best performance. As discussed above, the EnergyPlus simulation program covers a wide range of systems analyses of the building design and the list of its capabilities is still growing. Notwithstanding the apparent ingenuity of EnergyPlus, other performance simulation programs, such as the building regulations, building codes, design standards and cost analysis, could also be linked to the evolutionary design system.

A significant amount of descriptive and typological research has been carried out on the pedestrian environments where the density and direction of movement is exemplified through metric, geometric and topological models. Mathematical models have been developed to predict human behaviour in different scenarios. The systems and methodologies of

pedestrian environment dynamics have both broadened the horizons and multiplied, because they are adapted for planning other environments (Zacharias, 2001). In recent years, numerous studies have examined the behaviour and properties of pedestrian traffic on streets and inside buildings using computer simulations (Guo, Wong, Huang, Zhang & Lam, 2010; Hoogendoorn & Daamen, 2005; Yang, Zhao, Li & Fang, 2005).

The fact that human factors in designed environments predispose and influence the design solution means that they act as a major prejudice on design decisions and outcomes. The potential of the animation in architectural design applications as a key supporting element in the actual design process has not yet been fully acknowledged or used. Currently, animation is most often used in architectural design for visualizations (Dollens, 2006). Animating the architectural design holds great potential, however, in that the building's occupants' behaviours, reflecting the functional requirements expected and desired from the building, could be simulated and animated, then fed into the design system as design constraints fitness functions. Other human factors can also be imbedded, to govern the evolution of the design solutions. Clearly, not all human factors would be easy to digitally interpret, because the core differences between the closed logical systems of the digital environment are still at odds with the open logic system of human knowledge (Flanagan, 2005). However, the author believes that the gap is being narrowed as indicated and supported by evidence, such as the previously-mentioned example of the success gained in the pedestrian behaviour simulation.

Ideally, with regard to the proposed design system, the expectation is to have a multi-domain integrated-building performance simulation that provides the designer with a user-friendly interface together with the essential capabilities that would allow, for example, the rapid evaluation of alternative designs, for the design to be a rational decision-making process, for incremental design strategies and robust problem-solvers for nonlinear, mixed and hybrid simulations (Augenbroe, 2002).

5 Demonstration and Illustration

In this chapter, the demonstration and Illustration of the proposed Hierarchical Evolutionary Algorithmic Design (HEAD) System is discussed. This part of the research project falls into the 'developmental' stage of the overall research project and represents the synthesis of the design method that was conceived as a result of the 'conceptualisation' stage. Based on the in-depth study carried out into the development of the conceptual design method (discussed in Chapter 4 Conceptual Design Method), the system architecture is developed as a blueprint for the eventual implementation of the proposed design system in the 'Evaluation' stage of the overall research. Because the actual implementation of the proposed design system is not within the scope of this research project, the testing of the validity of the theoretical basis and the feasibility of the HEAD system is undertaken by running an illustrative paper-based simulation, with the key aspects being tested against a standardised building type.

5.1 HEAD System Architecture

5.1.1 Introduction

The design of buildings is a complex process involving not only the search for architectural design solutions that are aesthetically pleasing, but also the search for design solutions that answer the building's requirements from the perspective of multiple disciplines within the design and construction fields. In recognition that every design problem has a number of overlapping and distinct requirements and constraints that the designer needs to attempt to solve, the proposed HEAD system seeks to take a holistic approach to the design problem. The proposed HEAD system, based on this holistic approach, is believed to be capable of supporting the management of the multiple tasks and requirements of each discrete design problem. It is envisioned that taking such an approach will allow for the production of 'balanced' design solutions that consider all the various performance requirements of the building without losing sight of the need

for a single design solution that answers the building's requirements as a whole.

The proposed HEAD system uses the potential of genetic evolutionary algorithms. The benefits of genetic evolutionary algorithms can be used not only as the main engine for the generation of design solutions, but can also be used to optimise the successfully-generated models. This optimisation of the successfully-evolved models, run by genetic evolutionary algorithms, will be carried out in response to the simulated environment of the building that takes into consideration both the internal and external constraints of the design problem.

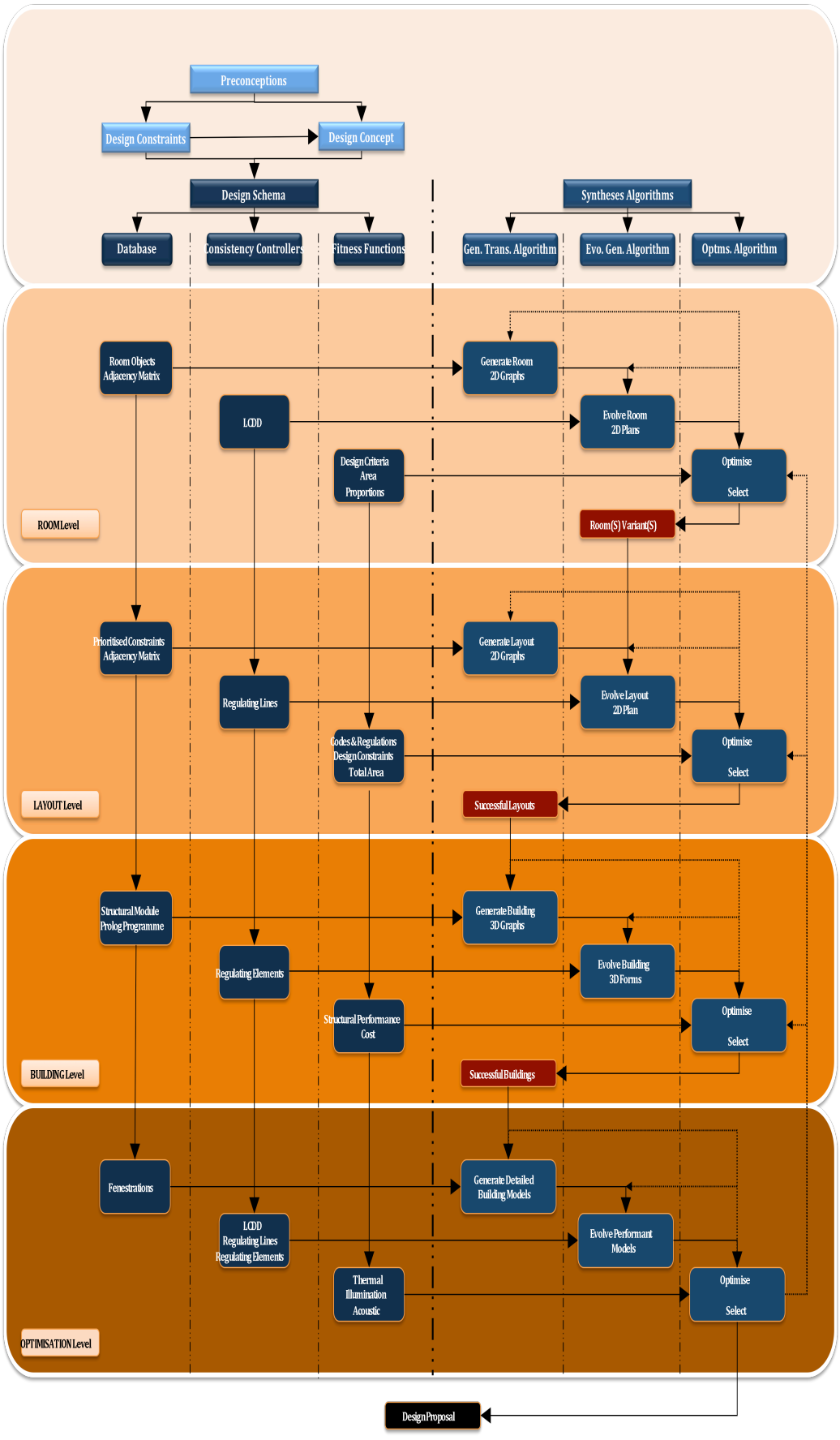


Figure 5-1 HEAD System

The HEAD system consists of two main components, the design schema and the Synthesis Algorithms. The design schema component is the designer's input stored in a database, and is intended as a means of maintaining design integrity and also provides the basis for any decisions that need to be made on the design that emerges from the second component. The Synthesis Algorithms component, equipped with generative evaluation and optimisation algorithms, directs the generation process of the emerging designs from one state to the next.

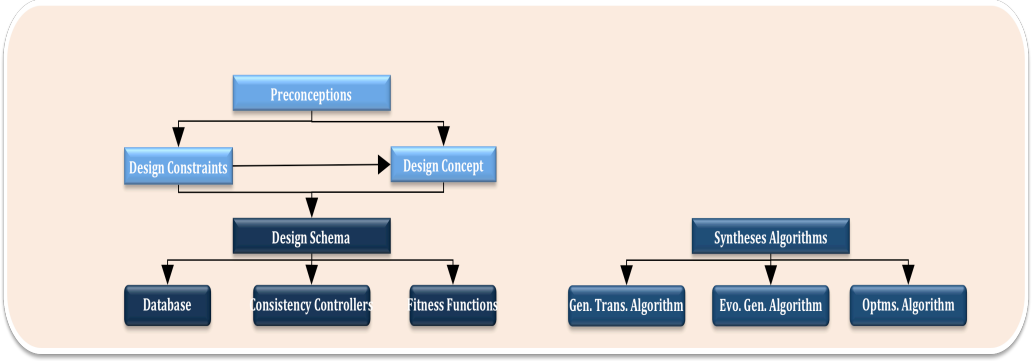


Figure 5-2 HEAD System components and sub-components

The design schema component is divided into a database sub-component and a consistency controller sub-component. The database sub-component includes all the objects and elements that constitute the building and the topology-geometry relationship between each component. The consistency controller sub-component ensures a regulated design process and administers the part-whole relationships, design hierarchy, scaffolding of the design process, the structuring of sub-problems and the structuring of the design parameters.

The Synthesis Algorithms component is divided into four hierarchical levels, room, layout, the building and optimisation. At each level, the generated solutions are required to achieve an expected goal. All levels of the Synthesis Algorithms consist of a generative transformational algorithm sub-component, an evolutionary genetic algorithm sub-component and an analysis, evaluation and selection processes sub-component. The fitness functions are set by the designer's interpretations of the design constraints in addition to the simulation of the internal and

external design environments that are assigned to the relevant levels. These fitness functions are used by the genetic evolutionary algorithms and are elaborated in sets of targeted sub-goals. The sub-goals are interpreted from the project's design constraints and the desired multi-disciplinary performance for each generated design state.

The HEAD system functions as a dynamic design methodology that processes the outputs, performance goals, optimisation and controls in an integrated way to allow for an overall design solution to be found for the building as a whole. To obtain a clearer and more accurate understanding of the design problem, the dynamic design methodology of the proposed design system initially divides the design problem into sub-problems and the building design task into different sub-tasks. As the various sub-problems and sub-tasks are processed through the hierarchical structure of the Synthesis Algorithms component, they are amalgamated into a unified design solution, incorporating all aspects of the design problem, which is then optimised in the final level of the system.

Because building design is a 'wicked problem' (reference), the initial runs of the HEAD system will lead to successive refinements of the design schema as the designer recognises undesirable solutions that can be removed by improvements. The evolution of the design representation and the analyses supports a closer analysis of the design problem within its context.

In the HEAD system, the design solution is generated and formed in response to the project's simulated environments. This is a departure from the existing performance-based design methods that use an analytical simulation of the previously-generated design solutions. The design solutions generated by the HEAD system are thus unique to that specific design space, which means that the design solutions unsuited to the specific project's environments are not produced, effectively eradicating the wasted time involved in generating and analysing every possible design situation for every combination of the design variants. The HEAD system could allow

for automatic adjustments and the development of the design solution during the actual generation and formation process.

The HEAD system interface completes the system and facilitates communication between the designer and the computer. Mediating between the real world and the symbolic structure of the system, the HEAD system interface:

1. Allows the designer to interact with it by supplying or correcting information that might be out-dated or inadequate;

2. Directs the design process by allowing the designer to make design decisions when appropriate.

Supports designer interactions within the design process by allowing the designer to intervene at two levels — by interacting with a selection of designs for further processing or, at the global level, by modifying the requirements that generate the seed designs and the fitness functions that drive design evolution.

5.1.2 Design schema

As discussed earlier, in Section (4.6), the Design Schema component of the proposed design system is a 'project-specific' process-oriented design model or framework for various types of projects. The design schema will be formalised as an executable process-oriented design approach that places a high emphasis, recognition and value on the aesthetics of the process and the organisation of the designed product. The design schema would, in effect, help the designer maintain his creativity throughout the design process by allowing the designer's preconceptions to be integrated into the interpretation of the design problem.

All design problems have a complex mix of multiple requirements and constraints that a designer must try to solve and satisfy. One of the first steps a designer takes when faced with a design problem is to formulate an initial design idea or concept that will aid him in untangling the web of

conflicting design needs and limitations. This initial design concept is formed out of the designer's perception, interpretation and subsequent prioritisation of the design requirements and constraints and is used by the designer as a guide to help focus his efforts in finding a solution for the design problem at hand. It is not uncommon for this initial design concept to develop into the eventual design solution. This research suggests that because the designer's perception, interpretation and prioritisation of the design problem(s) is clearly based on his preconceptions, especially his guiding principles, the designer's preconceptions can be seen to greatly affect and shape the design process through the capture of the design concept, thus enabling the designer to maintain his creative input into the design solution.

The values and the aesthetics of the organisation of the designed product and the actual design process itself can be further personalised and given greater emphasis and recognition by following the tectonic design approach introduced earlier (Section 3.2.3).

The Design Schema component of the proposed system provides the designer with the ability to create a unique (both to himself and to the specific design problem at hand) design model that he can use in his efforts to solve each discrete design problem. The designer's preconceptions will respond to, affect and influence the constraints of each specific project, in a unique and personalised way.

The design concept, primary generators and design schema are seen as design tools developed by an architectural designer to aid in the search and generation of design solutions. As discussed earlier, given a design problem (brief) a designer must tackle the specific design requirements and limitations of that design problem and develop a personalised and unique design concept to guide his efforts in finding a design solution. The uniqueness of each design concept, and thus the influence of the designer's creative vision, is further personalised by the designer's prioritisation of the design constraints according to his personal guiding principles.

The actual conceiving and formulation of the design concept that will guide the designer throughout the design process is a personal process and unique to each designer. The HEAD system does not attempt to understand, limit or enhance the creative processes of the designer. However, at the point at which the designer has formulated a personalised design concept, the proposed design schema is introduced into the process. This is when the design concept is required to be abstracted into a design schema that will then be used as a model or framework for the design process. Tranlsating a personalised design concept into a design schema that can be inputted into a computer is challenging, this step cannot be ignored. In the HEAD system, the designer's unique and personalised response to a unique set of design problems, which is formalised in the design schema, and the process-oriented approach used to solve those design problems, both play a crucial part in attempting to maintain the designer's creativity within the HEAD system.

The Design Schema component of the HEAD system reflects the designer's creative vision and understanding of the design problem and the overall configurations of the various elements and components that constitute the building as a whole. The Database sub-component of the design schema comprises the symbolic representational structure of the building elements, components and configuration that constitute each design state, while the consistency controller sub-component of the design schema helps to maintain the internal uniformity of all the elements, components and configurations.

The design schema feeds each level of the Synthesis Algorithms component with a set of data that accumulate from one level to the next (that is, the data used by the room level is added to the additional data assigned for the layout level and so on). This means that all the data used to evolve a design state at the end of one level becomes available to the next level for the propagation of new generations and for the eventual evaluation of that generation. Under the genetic algorithm concept, this accumulative data set would offer a direct means to the system for optimisation processes at the

active level which, in turn, means that unsuccessful solutions may not have to be immediately deleted, but rather, could be either optimised or even looped back to a previous level for more major optimisation.

5.1.2.1 The design schema database

The design schema database stores the knowledge of the design state that is needed to achieve the solution state targeted by each level of the Synthesis Algorithms component. For the room level, the knowledge stored is related to the functional requirements and relationships between each space and comprises the representation of objects, forms and properties. The planner module set for the permutations of rooms is also stored at the room level database.

For the layout level, information related to the design requirements, criteria and constraints is stored. Information pertaining to the organisation of the spatial relationships between the rooms, such as the overall configurations of the building and adjacency requirements, are also stored in this level. For the building level, data relating to the structural system and the building's envelope is stored. The properties and order of the structural components, together with a Prolog prescription of the overall building configurations is also stored for the building level. The optimisation level deals with all the information used by the first three levels to evaluate the overall performance of the generated models and to check the validity of the changes advised for optimising a solution to resolve a particular performance issue.

5.1.2.2 The design schema consistency controller

During the permutation of a design state by the generative transformational algorithm sub-component or during the propagation of new generations of design states by the evolutionary genetic algorithm sub-component, the consistency controller sub-component ensures a regulated process by administering the part-whole relationships. The consistency controller sub-component provides general constraints for those sub-

components to comply with throughout the generation processes at each level of the Synthesis Algorithms component.

The orders and the references of the consistency controller sub-component are a 'designer input' intended to provide guidelines to regulate the generation process in a non-arbitrary structural relationship. The room level consistency controller is mainly the LCDD geometric decomposition (Section 4.6.4). The layout level consistency controller is the generated regulating lines (Section 4.6.3). The building level consistency controllers are the regulating elements governing the structural form module (Sections 4.6.3 and 4.6.4.2). The optimisation level uses the previous levels consistency controllers, as this level deals predominantly with the overall performance of the produced solution states.

5.1.2.3 Fitness functions

The simulation of the project's environment (both external and internal) is introduced into the proposed design system as a design tool that will support the generation of the design solutions. The notion is that the simulation programs can be linked as fitness functions that will assess each individual of an evolved generation to the Evolutionary Design Systems. The successful, surviving solution with the best performance would then be selected. The simulation program intended for use in the proposed design system is the EnergyPlus simulation program (as discussed in Section 4.8.2), which is capable of running a variety of systems analyses of the building designs. However, the evolutionary design system could also be linked to other performance-simulation programs, such as the building regulations, building codes, design standards, structural performance analysis, pedestrian behaviour patterns and cost analysis.

Room level fitness functions take the form of area and proportions in addition to the design standards and the design criteria. Layout level fitness functions are the building codes and regulations and the building design constraints, in addition to the total built area and the adjacency requirements. Building level fitness functions are the structural

performance analysis and construction costs. Optimisation level fitness functions are environmental performance programs (specifically EnergyPlus) and a pedestrian behaviour patterns analysis systems (Guo et al., 2010; Hoogendoorn & Daamen, 2005; Zacharias, 2001), in addition to all the previous levels' fitness functions.

5.1.3 Synthesis Algorithms

The proposed architectural design system acknowledges and is designed to incorporate the notion that the design problem cannot be 'comprehensively stated', 'require[s] subjective interpretation' and 'tend[s] to be organised hierarchically' (Lawson, 2006, p. 120). The HEAD system is based on a strategy for solving design problems that involves the breaking down of the design problem into manageable sub-problems. The Synthesis Algorithms component is thus divided into four levels so that it can respond to the hierarchical decomposition of the design problem into multiple sub-problems. Each level of the Synthesis Algorithms component is thus able to tackle issues of the design problem sequentially.

The Synthesis Algorithms component controls the solution-search and design generation processes to achieve a series of goals and sub-goals at each level. Goals and sub-goals are formulated in a hierarchical structure so that they can be dealt with at the appropriate levels. The discrete levels in this hierarchical structure are arranged in a sequential order so that the relevant information needed for the effective running of the processes at each level is available when necessary. For instance, the layout level will require information from the room level, so the room level algorithmic processes are run before the layout level processes. At each level, a set of goals and sub-goals must be achieved through the processes of generation, evaluation and optimization. The outcome of each level comprises the information needed for the operations of the next level. Goal achievement is thus indicated by a transition from one phase (or level) to the other, because no transition can take place until the goals and sub-goals of the preceding level have been satisfied.

The HEAD system is not seen as a complete departure from the traditional design process, because the design solution produced by the system would undergo a number of stages of abstract representation, much like traditional design processes. This can be seen in the way that the Synthesis Algorithms component is set out in a hierarchy that relates to the inter-dependence of the design process, with the designer making a decision on the solutions to the sub-problem generated by the first level of the system before moving onto the next level. The chosen solution(s) from a lower level would then serve as the input into the next level of the system, together with any additional information needed for tackling the problem at the next level. Resolution of goals that impact lower levels are achieved by iterations over the range of proposed solutions, for example, selection of a wall as a structural wal impacting on acoustic performance.

5.1.3.1 Synthesis Algorithms Subcomponents

The Synthesis Algorithms component, working in a hierarchical sequential order, will achieve a sequence of goals according to the decomposition approach that the designer has taken to tackle the design problem. The Synthesis Algorithms component is made up of three subcomponents that operate on each level of the HEAD system. These are the generative transformational algorithm, the evolutionary genetic algorithm and the analysis, evaluation and selection subcomponents. The generative transformational algorithm subcomponent is responsible for the permutation of the initial generation that is needed for the evolutionary algorithm to operate. The evolutionary genetic algorithm subcomponent is in charge of the propagation of generations using genetic operations. The analysis, evaluation and selection processes subcomponent is responsible for the assessment of the produced generations and determines how well the criteria and the constraints that were prescribed by the fitness functions were met.

The first three of the Synthesis Algorithms component levels are built with a generative capability that can feed the evolutionary genetic algorithm subcomponents with the initial generation that will start the evolution

process. The first three levels can be recognised by the optimisation level as the generative subcomponent that is implanted in each of them. The optimisation level is considered to be the analysis and evaluation subcomponent that is implanted in the first three levels. The successfully produced and accepted design models generated by the first three levels feed the last level with a fitter generation, which will then be further evolved to produce more optimised generations based on the performance.

5.1.3.2 Synthesis Algorithms Levels

Each of the four Synthesis Algorithms component levels is responsible for tackling its designated sub-problems, based on the data and controllers associated with it. Each level generates solutions that are analysed, evaluated and optimised and which are subsequently fed into the next level for solving the next sub-problem. The accumulative nature of the hierarchical levelling of the design schema and the Synthesis Algorithms components, where the data of each level is made available for the next level, provides a reference for the decisions made in the production of the design states within a previous level. This does not mean that the later level is required to resolve conflicts occurring as a result of the previous decisions. The compilation of the data regarding the decisions made at each level offers a record of the basis on which the decisions at the previous levels were made; this means that a greater understanding of the design solutions can be gained. In addition, if the conflict can be resolved by means of prioritisation, the system will make that decision and go on; otherwise, using the accumulated data as a reference, the conflicting results can be looped back to the responsible level (the level at which the conflicts arose) for further optimisation. In effect, this means that the whole problem (with its multiple conflicts) does not have to be tackled simultaneously at one single level.

5.1.3.3 Goals and Sub-goals

The goals and the sub-goals constitute the design plan and are comprised of both a hierarchical and a sequential ordering. Goals represent the accomplishment of the phases reflected through the HEADS levels that

follow each other (room design, followed by layout development, building design, and so on), while sub-goals elaborate the requirements that make up each goal. The sub-goals are interpreted from the design constraints and are added to the assigned evolutionary algorithms' fitness functions.

Sub-goals are expressed in prioritised weighted criteria to be satisfied by the design state targeted by each sub-component in each level. Such expression does not impose the organization of the design state itself, rather, it advises the relative qualities and the least acceptable trade-offs. Some combinations of the achieved sub-goals that may not be found to be optimal could be found to be adequate for satisfying the goal itself. This is especially true when the fulfilment of certain sub-goals is influenced by the characteristics of the emerging design state.

The prioritised-weighted criteria structuring of the sub-goals provides a foundation for rich and diverse permutations from one design state to the next. Although the sub-goals within each of the Synthesis Algorithms component levels are sequential, not all of them are expected to be completely achieved to satisfy a given goal. The sequential formalisation of the sub-goals guarantees the adequacy of the information fed into each subcomponent, with the accumulated achievements ultimately marking the success of producing the design solution state as a whole.

Every decomposed sub-problem requires the achievement of different goals and sub-goals. Thus, each level of the Synthesis Algorithms component requires different sets of goals and sequentially-set sub-goals. Therefore, the development of the modular sub-goal-sets that can be substituted and called on to optimise a particular problem as needed. The modular sub-goal-set facilitates the looping back to an earlier design stage if and when conflicts rises that need to be optimised at the preceding level.

5.1.4 System operation

5.1.4.1 Room level

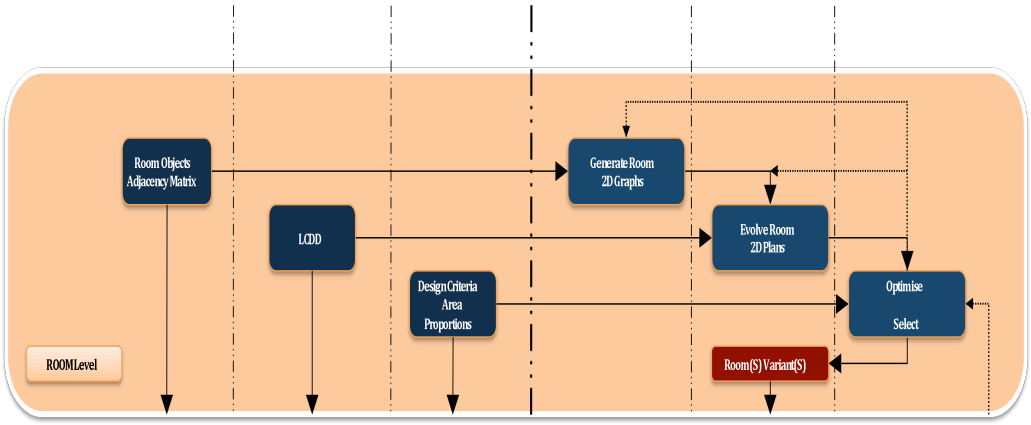


Figure 5-3 Room level

The room level (Figure 5-3 Room level), as the name suggests, is responsible for producing different variants for each room based on its constituent parts. The room component is encoded with its special requirements and its relationship to other components. The adjacency relationship between the room components is translated into graphs and fed into the generative transformational algorithm sub-component to be permutated into the initial generation. This is needed for the evolutionary genetic algorithm subcomponent to propagate evolving generations.

When part of the solution is predefined, the designer's creative expression is not given free reign, because the system will only search for solutions from within the space of the constituted building, based on the predetermined solution. If the genetic algorithm is allowed to generate a successful variety of parameterised spaces according to a predefined set of rules and assigned fitness functions, rather than the solution itself, the system can choose from a variety of different shapes and sizes (for each space) to generate a variety of layout plan configurations.

The room level algorithm is encoded with the architectural design standards of the spatial requirement for the relevant activities hosted in each room to fulfil its designated functional requirement. The algorithm generates a variety of alternative room layouts with different shapes, areas

and dimensions. The parameterised variants of each room are dynamically adjustable to fit within the different layout configurations that are generated, based on the adjacency requirements at the layout and building levels of the Synthesis Algorithms component of the HEAD system.

5.1.4.1.1 Room level database

Buildings are comprised of any number of spaces that host the variety of functions the building is designed to serve. One function of the main early activities in the architectural programming of a facility is to determine the required spaces for the building, and the function and occupant numbers that will be hosted in each. The spatial requirement of each room, for it to be capable of hosting the anticipated activity and also its relationship to the rest of the building in terms of its adjacency requirement, is generated out of this initial information. As a result, the total size of the building and the complexity of the required systems needed to support the overall performance required from the building can be initially stated, at the very least, in the mind of the designer. However, in the early stages, this does have some degree of flexibility.

5.1.4.1.1.1 Objects

Some of the flexibility is inherited from the design standards for the spatial and area allocations for an occupant conducting a specific activity. Each room consists of allocated spaces together with special pieces of furniture or fixture where an activity is conducted. Designers, using a bottom-up design strategy for layout planning, decompose the spaces to their constituent elements to be able to explore the wider variety of areas and dimensions for each space, rather than working with a fixed area and dimensions. This approach provides the flexibility and freedom for the designer to explore different varieties of room shapes, area and dimensions, resulting in a wider range of overall layout configurations of the whole building, and thus shape and form.

In the HEAD system, each room is decomposed into its constituent parts according to its functional requirements. Using qualitative modules and

spatial categorisation for the architectural entities (Bhatt et al., 2009), which were discussed in Section (4.6.2.1), each object is analysed, based on its spatial requirement with reference to the architectural design standards and the room design criteria. Consequently, each object with its established object space, operation space and functional space is allocated with its spatial requirement (for example, the adequate space and dimensions required for each object).

5.1.4.1.1.2 Object adjacency relations

The relationships between each object and the other objects within the same room are translated into a weighted adjacency matrix. The adjacency matrix (discussed in Section 4.6.2) provides alternatives and therefore a number of different graphs can be generated for each room.

5.1.4.1.1.3 2D LCDD module

The designer creates a two-dimensional Lowest Common Design Denominator (LCDD) module (discussed in Section 4.6.4.1) with the assigned transformation rules governing its permutations in the generation process of variants for each room.

The LCDD is a basic geometric seed that articulates the designer's interpretations of a part that can be manipulated and arranged to eventually constitute the whole. The LCDD is defined with a variable set of parametric rules and mechanisms, which leads to a compositional formation when applied. Starting at the room level and moving through the remaining levels, the LCDD permeates and mutates the 2D LCDD module or the form module into the design states. Ultimately, the use of the LCDD is expected to result in regulated, coherent and consistent generations of design solutions.

5.1.4.1.2 Room Level consistency controller

5.1.4.1.2.1 LCDD

The two-dimensional LCDD module regulates the room generation to avoid generating rooms with arbitrary dimensions that will make it harder to place adjacent rooms together. Room generation following the 2D LCDD

module establishes the grounds for the later stages of layout planning and allocating the structural components to generate the building form for the evolved layout.

5.1.4.1.3 Room level goals and sub-goals

Clearly, the room level goal is to generate acceptable variants for each room. The variety of the shapes and dimensions of each room variant enriches the next level with many alternatives of the same room to choose from during the space allocation process.

The sub-goals indicate the achievement of this level's goals and are expressed through the compliance of each generated variant with respect to the spatial requirements and the relations between the objects within the room itself. In addition, the generated variants marked as successful should comply with the room design criteria and the architectural standards.

5.1.4.1.4 Room level fitness functions

5.1.4.1.4.1 Architectural design standards

The human form is commonly used as the basis not only for generating the architectural design standards, but also for manufacturing the pieces of furniture or a building fixture. The obvious reason is that most buildings are usually occupied and used by humans, except for special purpose buildings, where there is a mixture of types of users (and uses), for instance, in factories or farm buildings and so on. The area and dimensions needed for an activity conducted in a room and for the furniture and fixtures together with the area and dimensions for using them is set in a range of a maximum and minimum requirements, with the most common values usually lying in the middle. This range allows for wider variations of the rooms to be produced during the assembling of the room constituents. In the proposed design system, the architectural design standards of the relevant building components and spatial requirements are inserted as a fitness functions for the assessment of the generated designs.

5.1.4.1.4.2 Design criteria

Each room has design criteria requirements according to its functions and the activity it is build to accommodate. The criteria of each room type reflects the past experience of similar activities being conducted in certain rooms in addition to the needs descended from the design standards and the social and economical values. The room design criteria for each room, which constitutes the building being designed, is encoded and inserted as a fitness functions into the proposed design system for the assessment of the generated design.

5.1.4.1.4.3 Area

The area of the generated room indicates the adequate space to accommodate the required functions of the room. Oversized rooms result in a waste of space and higher costs, while undersized rooms jeopardize the sufficiency to host the spatial and functional requirements.

5.1.4.1.4.4 Proportion

The dimensions of the generated room perimeters reflect its proportions. Guided by the planner module generated by the designer, a mathematical formula of a geometric proportion can be used to evaluate the aesthetic value of the generated room. Formulas of the proportions, such as the golden ratio or the square root of two, can be used for two-dimensional proportions. Following Palladio, the sixteenth century Italian architect's methods, elucidated Pythagorean arithmetic, geometric and harmonic means can be used to calculate the height proportions of a room (Palladio, 1997).

5.1.4.1.5 Room level generative transformational algorithm

The subdivided spaces of the room components' spatial requirements (object vs. the operational vs. functional) indicate which space of a component is allowed to overlap with the space of another component. For instance, the functional space of a component is usually allocated for

circulation purposes and thus can be used for the same purpose by another component's functional space. This provides flexibility during the packing process.

The graphic representation of the relationships between the different room objects is translated into a two-dimensional dual graph of the relevant geometrical properties of each object. Therefore, in addition to the analysis of the objects' spatial requirements, the room's spatial requirements (area, dimensions) can be established. The planner module selected by the designer can be multiplied and arranged in several ways to accommodate the dual graph representation of the room objects according to their relationships and thus the room's functional requirement can be met.

After the total area required to host all the objects in the room is known (after the packing of those objects has been done), the algorithm searches for the right arrangements of the planner modules that can accommodate the constituent parts of the room. The initial permutated room variants are based on the highest preferences of the adjacencies between the objects within the room.

5.1.4.1.6 Room level evolutionary genetic algorithm

Each adjacency requirement between objects in the room can be represented in several different graphs, including dual graphs. In addition, each dual graph representing the adjacency requirements can be adequately represented in several arrangements of planner modules. The evolutionary genetic algorithm propagates generations of room specimens through searching the solutions' space to find suitable arrangements of planner modules that can accommodate an acceptable level of adjacency fulfilment.

The relationship established between the dual graph of the room objects, and the arrangement of the planner modules hosting it, facilitates the evolutionary algorithm with suitable representations of the genetic operations to take place. In other words, the genetic operations can take place between the dual graph of one parent and the arrangement of

planner modules of the other parent or within either the dual graph or the planner modules arrangement of the two parents.

5.1.4.1.7 Room level analysis, evaluation and selection processes

The evolving generations are analysed, evaluated and assigned with a score that reflects their fitness. The room fitness functions are the area, perimeter and proportion, in addition to the compliance with the spatial requirements.

The variants of each room with the highest score are selected and stored into the layout level database, ranked by score. The successful room level variants and their ranks can then be used for the space layout planning.

5.1.4.2 Layout level

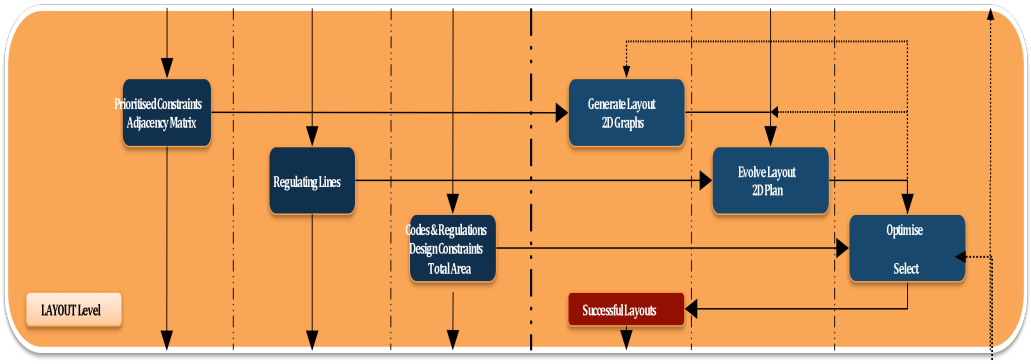


Figure 5-4 Layout level

The layout level, (Figure 5-4 Layout level) is responsible for producing the successful building layouts in response to the design constraints and the spatial relation requirements. The room variants generated from the room level are searched to find the most suitable combinations, based on the spatial adjacency requirements and with respect to the defined design constraints. The regulating lines are used as the basis for the layout planning process with reference to the site and its surroundings.

Using circuit graphs for each configuration of the building generated by the adjacency requirements, as entered by the designer, the algorithm searches the produced room variants of each room, starting with the highest score

room variant, and packs them accordingly. Each generated graph and consequently its packing layout configuration is assessed, based on the assigned fitness functions, and the graphs with the highest score are fed into the evolutionary algorithm as parents of the initial generation. Genetic operations then take place between the selected parents to evolve the next generations. The successful layout configurations with the highest score are then selected and fed into the next level.

5.1.4.2.1 Layout level database

5.1.4.2.1.1 Constraint prioritisation

The internal and external design constraints that the designer must deal with are both varied and numerous. Very often, the constraints conflict with each other and this can lead to unforeseen problems. The issue of the conflicting constraints can seem to present an overall problem that has no possible solution. One way a designer can tackle the problems created by these conflicts is by prioritising the constraints. By prioritising the design constraints, the designer may not be able to satisfy every aspect of every design requirement, but will be able to find the most satisfactory solution to the conflicting problems of the design.

The project's identified constraints are encoded in a descending order according to the relative priority assigned to them by the designer. The prioritisation of the design constraints (discussed in Section 4.4.1) is supports the resolution of conflicting constraints. In addition, prioritising the constraints means that a satisfactory solution can be found, even if it is not fully compliant with every constraint.

This process of prioritisation of the design constraints can be based on the needs of the client, the users and society but ultimately, it is very greatly influenced by the designer's guiding principles. It is most likely that different designers will prioritise the same set of design constraints differently. In his attempt to fulfil the project's goals as effectively and as fully as possible and by making decisions about the importance of each aspect of the constraints he faces, the designer is given another opportunity

to inject his personal creativity into the design schema and ultimately into the design process.

5.1.4.2.1.2 Spatial adjacency requirements

The designer sets the adjacency rules and requirements for the design, based on the spatial and functional relations between the building spaces. The spatial relations and the adjacency requirements between the rooms need not always be strictly fixed, as in cases where there is some degree of flexibility or, in some cases, no preference at all. This feature also presents the designer with the opportunity to express his creativity and allows him to develop data on weighted adjacency requirements to host multi-scenarios of room adjacencies that will result in the generation of a variety of overall building configurations.

5.1.4.2.2 Layout level consistency controller

5.1.4.2.2.1 Regulating lines

The regulating lines include the axes of symmetry, centre or rotation, alignment axes, diagonal proportion lines, bounding lines, building lines, setbacks and so on are implemented in a regulated design process and design schema, to administer the part-whole relationships, design hierarchy, scaffolding the design process, topology-geometry relationship, structuring sub-problems and the structuring of the design parameters.

Regulating lines are generated from the properties of the site and its surrounding urban settings. The designer, while conducting the site planning and analysis, identifies the elements, including the location, neighbourhood context, size and zoning, legal elements, natural physical features, man-made features, circulation, utilities, sensory, human and cultural, and climate components. The physical, biological and cultural attributes of the site not only present themselves as major design constraints, but also enforce some directions for the designer to follow. For the building to be integrated within its environment, the designer must respect the site's features by not imposing any foreign solutions drawn from unrelated sources. Sites are usually a rich source for the designer to

generate regulating lines that might help their primary generators to capture a design idea and formalize their solution around it.

The regulating lines are the foundation guidelines for positioning the building within the site and for determining the building borderlines, divisions and sub-divisions, which are the source for the organisation of the comprehensive formation of the architectural arrangement.

5.1.4.2.3 Layout level goals and sub-goals

The goal of the layout level is to produce alternative successful layouts for the building's configuration. The surviving propagations with the highest fitness function score are passed onto and enrich the next building level with a variety of layouts that can be readily constructed into three-dimensional models.

The achievement of the sub-goals of the layout level is indicated by the achievement of the layout level goal that is, in turn, indicated by the fulfilment of the design constraints with the minimum trade-offs for layouts with a higher preference of adjacency requirements. In addition, the degree of compliance with the regulating lines and a coherent configuration of the rooms is also an indication that the goals and sub-goals of the layout level have been satisfied.

5.1.4.2.4 Layout level fitness functions

5.1.4.2.4.1 Building codes

There have been many successful attempts to codify building codes (Rosenman & Gero, 1985) and building design standards (Stahl, Wright, Fenves & R. Harris, 1983) for expert design systems. The encoded information from these codes and standards systems can be used for assessing the generated solutions' level of compliance with the relevant codes and standards. This can be achieved by linking the systems used for codifying the building codes and standards with the analysis, evaluation and selection sub-component of the Synthesis Algorithms component.

5.1.4.2.4.2 Design constraints

General types of design constraints affecting the layout planning, such as the adjacency requirements, spatial allocation of certain rooms, indirect spatial relations between space and so on, are encoded and inserted into the Synthesis Algorithms component according to the designer's prioritisations.

Because the adjacency requirements between the different spaces of the building are set in a weighted matrix, each generated configuration represents a level of preference that can be compared with the configuration of the highest preference. Additionally, the travelling distances that have the shortest travelling distances between the related rooms can be measured and compared with the generated layout.

5.1.4.2.4.3 Total area

The total area of the generated layout and its perimeter length are important factors for determining not only the cost, but also the building's exposure to the external environment. Although at this stage of the design, the environmental performance does not determine the successfully-generated solutions, these factors are important for the functionality and the cost assessments.

5.1.4.2.5 Layout level generative transformational algorithm

Graphs representing the highest possible adjacency preferences are generated from the adjacency requirements matrix. Starting with the highest in rank, the generative transformational algorithm searches the generated variants of each room to select the most suitable alternative that fits with the other adjacent rooms and which also complies with the regulating lines. Consequently, a dual graph with room dimensions is generated for each successfully generated layout.

5.1.4.2.6 Layout level evolutionary genetic algorithm

The permutated configurations of the building layouts, represented in the dual graphs, with imbedded integer numbers indicating the selected room

variants, are the initial generation for the evolutionary algorithm to start the propagation process.

Because the permutated layouts are represented by the dual graphs and the integer string that refers to the selected room variants, the genetic operations take place between the graph representation of one parent and the selected rooms' variants of the other parents or within either the dual graph or the room variants of the two parents.

The regulating lines play an important role for guiding the propagation process by providing a referent grid system during the search for the most suitable room dimensions and shapes among the generated variants. The evolutionary genetic algorithm is responsible for finding layouts that are aesthetically consistent with the help and reference of the regulating lines.

5.1.4.2.7 Layout level analysis, evaluation and selection processes

Some generated solutions might need to be optimised to produce a fitter solution because another room variant might fit better in the generated layout. The system could replace the unsuitable variant with a more appropriate one from the database of the generated room variants or even loop back to the room level to generate the required room. The optimised layout is then re-evaluated before it is considered unsuccessful and deleted.

In addition to the evaluation of the aesthetic value of the generated layouts, measured by their level of compliance with the regulating lines and also the fulfilment of the adjacency requirements, supplementary fitness functions are included in this level to evaluate the functional performance (for example, the total area of the building, building perimeter length and the cost).

5.1.4.3 Building level

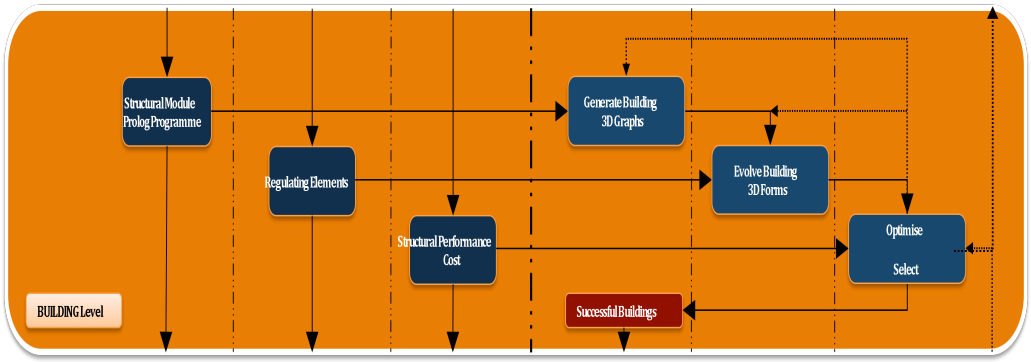


Figure 5-5 Building level

The Building level (Figure 5-5 Building level), is responsible for turning the successfully-generated layout configurations with the highest score into three-dimensional models by constructing the structural skeleton using the parameterised modular structural components of the chosen mode of fabrication. The Prolog prescriptions of the building and the regulating elements, together with the new properties of the evolved space layout configuration, set the basis for the expression of the entire structure. The structural support points are assigned according to the building fitness of axis, mode of fabrication and structural system.

Each layout configuration has the potential for producing a variety of models generated from the parameterised modular structural component. Each of the generated models is assessed, based on the assigned fitness functions and then the models with the highest score are fed into the evolutionary algorithm as the parents of the initial generation. Genetic operations are then carried out between the selected parents to evolve the next generations.

5.1.4.3.1 Building level database

5.1.4.3.1.1 Structural form module

The designer prescribes his perceptions of the overall building configurations using the Prolog definite clause grammars 'declarative' language (discussed in Section 4.6.1). The Prolog prescription is translated into a parse-tree representation where different grammatical strings of the

building configurations can be produced to govern the Synthesis of the building's structural components.

Both the 'facts' and the 'rules' governing the design problem are formalised by the designer, thus the solution that the Prolog program ultimately produces will incorporate the designer's personal interpretation of the design requirements, goals and constraints. By using definite clause grammars, the designer is able to set queries about which relationships are 'true' and thus, the solution found may not have been directly reached by the designer, but the solution-finding process can be directed by him.

5.1.4.3.1.2 Prolog drescription

The designer prescribes his perceptions of the overall building configurations using the Prolog definite clause grammars 'declarative' language (discussed in Section 4.6.1). The Prolog prescription is translated into a parse tree representation where different grammatical strings of the building configurations can be produced to govern the Synthesis of the building's structural components.

DCG, can be used to govern the interaction between the designer and the computer. The interaction between the human and the computer takes the form of the designer specifying the formal relationships between objects (or spaces that make up a building) using Prolog programming. Setting only very basic 'grammar rules' for a design problem, (or any problem for that matter) can lead to some implausible solutions however. The designer can avoid this by generating a syntactic parse tree, which represents the syntactic structure of a string, ordered and rooted according to a definite clause grammar. The designer, using DCG and parse trees is thus able to prescribe the spaces and relationships between spaces within the building using Prolog, is thus able to prescribe the spaces and relationships between the spaces within the building.

5.1.4.3.2 Building level consistency controller

5.1.4.3.2.1 Regulating elements

The regulating elements are the underlying structure for the massing configurations. The regulating elements include, in addition to the regulating lines themselves, points, planes and the volumes generated by the regulating lines and their points of intersections.

The regulating elements relate the massing configuration, organise the elements of the massing configurations, form the spatial sub-division within the masses and relate the massing elements to their constituents. Following the regulating elements as the basis for the expression of the entire structure of the architectural configuration, the generated layout of the spatial configuration is transformed by the parameterised modular structural component into a three-dimensional form.

5.1.4.3.3 Building level goals and sub-goals

The goal of the building level is to produce parameterised models from the successfully-generated layouts. The generated model will have sufficient information for conducting an environmental performance evaluation and would be optimised by looping the specific conflicting problems back to the responsible level.

5.1.4.3.4 Building level fitness functions

5.1.4.3.4.1 Structural performance analysis

Parametric design in parallel with the structural performance analysis software is linked to the genetic assessment as a fitness function and allows the design process to optimise the design model, based on the engineering data.

5.1.4.3.4.2 Cost

There are many methods for calculating the building construction costs, from the rules-of-thumb to a detailed scheduling of the bill of quantities for all the building systems and labour costs. At this level of the preliminary design phase, where most of the design decisions are not yet finalised and thus the quantities and information of the building systems, materials and

so on are not finalised, it is logical to use a general formula for the calculations, such as, for example, the construction cost of a square metre of a building.

5.1.4.3.5 Building level generative transformational algorithm

Based on the Prolog prescriptions of the building configurations that are translated into a parse-tree and consequentially, the grammatical strings to govern the Synthesis of the building structural formation, the algorithm, with reference to the regulating elements, allocates the insertion points and lines for the three-dimensional objects of the building, such as the structural supporting components and the internal and external walls.

The structural components with their transformational rules and relations governing the assembling activities have the potential to permutate a number of alternative structural skeletons for the same layout. Each permutated alternative for each successful layout is translated into a three-dimensional rectangular dual graph (see Section 4.6.2) forming the initial generation for the evolutionary algorithm to operate.

Once the Prolog program has generated the parse-tree, it is rendered as a dual graph. The dual graph is used as a graphical representation of the spatial organisation of the building's spaces according to the adjacency constraints between those spaces. The dual graph also incorporates the regulating elements and, as a whole, it is the visual representation of the designer's conception of the spatial configuration of the building. The spatial configuration represented in the dual graph serves as an automatable depiction that can be fed into the genetic algorithm system. This transfer into the genetic algorithm system is carried out by translating every dual graph into three-dimensional rectangular duals. These are then encoded into the genetic algorithm system with the expected performance criteria set to govern the evolution. The adjacency matrices and reproduction operators of the graphs can be topologically configured and encoded for use by the genetic algorithm system to allow for further exploration of the larger space of the graph topologies.

Using graph theory for the space layout planning is based on regulating the elements that are managed and directed by the regulating design process. The overall configuration of the design solution, governed by the Prolog description of the building, is also built upon the foundations of the regulating elements. In addition, the guidelines for the translation of the parse-trees generated by the Prolog grammar in a non-arbitrary spatial configuration are provided by the regulating elements. By relating the massing configuration and organising the elements of massing configurations, the regulating elements serve as the basis for the expression of the whole architectural configuration of the design in three dimensions.

5.1.4.3.6 Building level evolutionary genetic Algorithm

Each permutation of the building model for a successfully generated layout is represented in a developed three-dimensional rectangular dual graph embedded with grammatical strings of the parse-tree generated by the Prolog prescriptions of the building configurations in addition to the structural information.

The initial generation consists of the permutated parents with identical layouts, but with different structural configurations, and the parents with different layouts and consequentially different structural configurations. On the one hand, the genetic operations of the evolutionary algorithm take place, between the parents with identical layouts, within their structural configurations. On the other hand, the genetic operations take place on the entire permutated model for those parents with different layouts and structural configuration.

5.1.4.3.7 Building level analysis, evaluation and selection processes

Because some generated solutions might require optimisations to be made fitter, the system first tries to conduct the needed optimisation within the level with reference to the transferred data from the earlier level. If the problem cannot be resolved internally, the HEAD system then loops back to

an earlier level that is relevant to the particular issue to be resolved before considering the model as unsuccessful and setting it aside for deletion.

The regulating elements developed from the regulating lines from the previous level continue to provide the aesthetic reference and order for the generation of solutions. The evaluation of the aesthetic value of the members of the produced generations is based on the level of compliance of each successful member with the regulating lines and elements. In addition, the structural performance software is connected to the process to evaluate the structural behaviour of each propagated model.

As explained earlier, all the data and the fitness functions in one level are automatically made available and added to the next level. On top of the new added fitness function of the level undergoing propagation, the previous fitness functions are checked and re-assessed, especially when an optimisation takes place.

5.1.4.4 Optimisation Level

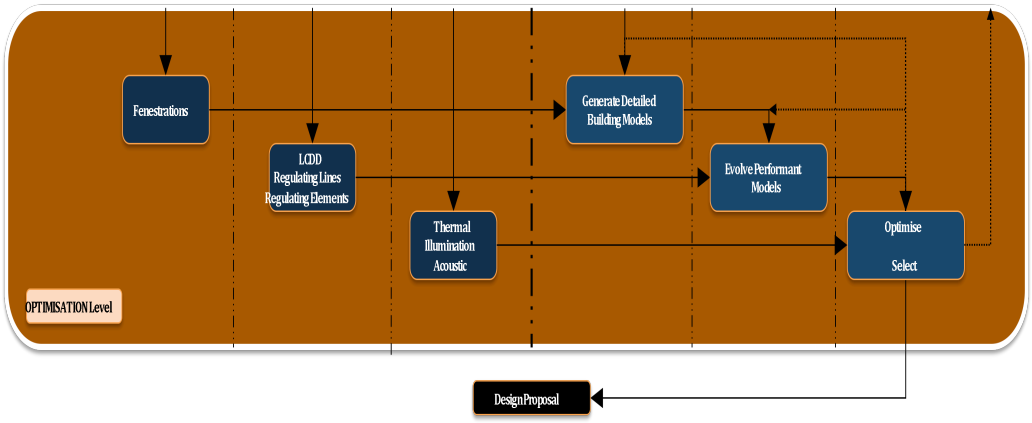


Figure 5-6 Optimisation level

The final stage of the HEAD system is the optimisation of the successfully generated design models, (Figure 5-6 Optimisation level). The main function of the optimisation level is to search the narrowed down design state space for the fittest design solutions. The looping back to earlier levels to optimise particular problems is expected to take place more frequently at this level. Nevertheless, the advantage of not having the HEAD system explore the whole design space, including the design states that are not

acceptable, is achieved with the hierarchical levelling and the decomposition of the design problem. This also allows resolution of global constraints.

Each level of the HEAD system is assigned with fitness functions that are either directly related to the particular level of design state (i.e. room, layout or building level) or generally, to assess as early as possible the optimal design states that will subsequently be generated out of the last level of the system. Energy, for example, can be inserted as a fitness function as early as the layout level, although it should be noted that the assessment would not be completely accurate at the earlier levels due, for example, to the fenestration not having been decided at such an early stage in the design process.

It is at the optimization level that the accumulated sum of all the fitness functions from all prior levels are combined with new fitness functions that cannot be assessed at earlier levels, the design solution as a whole is then assessed. In effect this means that more accuracy is achieved as the design state matures, until the optimal design solution is ultimately achieved.

The Evolutionary Design Systems, on the one hand, using a repeated cycle of genetic scripted code, are able to produce designs adapted to the simulated environment. Performance Based design, on the other hand, applies modifications to the produced design, based on its performance in an analytical simulation. The HEAD system develops a generative formation design system with two environments encoded into the system. The project's external constraints simulate the external environment of the project. The project's internal constraints simulate the internal environment and animate the desired performance of the project. Thus, the design would be generated in response to the simulated external and internal environments and also to the analysed desired performance.

Performance analysis on the chosen model is conducted to verify different performance aspects of the produced designs. Although available performance analysis applications are valuable tools, they do not normally

come with modification capabilities. To overcome this issue, when modifications are required or even to test the impact of design decisions, alternative design solutions can be fed back into an earlier phase by varying the chosen parameters.

There are no optimal solutions to any significant building design problem but rather, an infinite number of possible solutions (Lawson, 2006). Thus the HEAD system loops back to earlier levels allowing the system itself or the designer to make modifications on the successful selected solution.

5.1.4.4.1 Optimisation level database

In addition to all the database information accumulated through the previous levels, the optimisation level database is fed with the desired or expected performance of the building's internal performance. The environmental performance criteria (thermal, illumination, acoustic and so on) of each space in the building are usually defined according to the type of activity that the space is designed to host. The system analyses and evaluates each generated solution to determine whether it complies with the requirements or whether it needs further optimisation.

The types of activity to be conducted by the occupants within all the spaces of the building are animated to test the level of compliance in each generated room, and the whole building model, with the spatial functional requirements. The same generated graphs for each room in addition to the graph of the whole building configurations, where the occupants circulate from one station to the other according to a particular sequence, are used to demonstrate any or all of the building's activities.

Mathematical formulas predicting human behaviours in different scenarios are inserted into the optimisation level database to guide the simulation of the occupants' sequence of activities that are to be conducted throughout the building.

The data for the fenestration components are inserted in the optimisation level database to enable the environmental performance analysis of the

building. The designer selects the window types and shapes and parameterises them by setting up the proportional relations between the window variants.

5.1.4.4.2 Optimisation level consistency controller

The optimisation level abides by all the consistency controllers of each previous level during the permutation and the propagation processes to evolve solutions that fall inline with the references and orders from previous levels.

Because changes might be imposed on an object (from the design states permutated by the generative transformational algorithm subcomponent or those propagated by the evolutionary genetic algorithm) through the optimisation processes, the consistency controller subcomponent will determine which other objects, if any, will be affected, and how. The knowledge base of this subsystem is thus capable of navigating the relationship network, in addition to resolving such problems as compliance with the consistency controllers (LCDD, regulating lines and regulating elements).

5.1.4.4.3 Optimisation level goals and sub-goals

The HEAD system produces the design solutions that maintain a balance between the various performance requirements expected from the building by integrating the simulation tools at the earliest phases of the design process to evaluate performance and therefore effectively relate design decisions to the simulation results.

The optimisation level is responsible for achieving this ultimate goal through the successful propagation of the fittest generations in response to the various fitness functions of the external constraints, the internal constraints and the self- imposed constraints accumulated from all the previous levels. The sub-goals can be seen in the achievement of ‘satisficing’ the balance between the different interdisciplinary performances expected from the building.

5.1.4.4.4 Optimisation level fitness functions

The HEAD system uses the simulation of the project's environments as a tool in the generative processes used to find a design solution. That the simulation programs can be linked to the Evolutionary Design Systems as fitness functions, which will assess each member of the evolved generation. The selection of the successful surviving solution can thus be made on the basis of the best performance. The EnergyPlus simulation program (see Section 4.8.2) can be used for a variety of building design analyses. However, other performance simulation programs, such as the building regulations, building codes, design standards and cost analysis, can also be linked to the evolutionary design system for a greater coverage.

Descriptive and typological research on the pedestrian environments has led to the density and direction of movement being expressed through metric, geometric, and topological models and the prediction of the human behaviour in different scenarios has been shown to be possible using mathematical models (Guo et al., 2010; Hoogendoorn & Daamen, 2005; Zacharias, 2001). The potential for the animation of the architectural design thus holds great promise, because the behaviours of the building's occupants, reflecting the functional requirements expected and desired from the building, could be simulated and animated using the models mentioned above. These simulations could then be fed into the design system as design constraints and fitness functions. The evolution of the design solutions can also be defined and controlled by other human factors embedded into the system. Obviously, not all human factors can be digitally defined with ease and, although the gap between the closed logical systems of the digital environment and the open logic system of human knowledge has narrowed, it is still wide and requires bridging.

5.1.4.4.5 Optimisation level generative transformational algorithm

With reference to the regulating lines and elements and the proportional set-up of the LCDD planner module, together with the environmental performance criteria for each room, the generative transformational

algorithm inserts the windows for the global building fenestration requirements. Each of the generated models from the previous level, permutates several models with different fenestration solutions. The evolutionary genetic algorithm is fed with the initial generations of all the models and all their facade variations.

5.1.4.4.6 Optimisation level evolutionary genetic algorithm

The genetic operations occur between the parents of the same model but with different fenestrations or between the parents of different models and consequentially different fenestrations. The former case propagates the same sibling model but each with different facades. While the later case, propagates siblings of different models and consequently different facades.

While the search is under way to achieve environmentally performant models, it looks up and checks the functional suitability of the generated models to the occupant's activities conducted throughout the building by the simulation of each activity.

5.1.4.4.7 Optimisation level analysis, evaluation and selection processes

The optimisation level Analysis, Evaluation and Selection Processes subcomponent is in charge of the inner optimisation of any generated solutions needing minor optimisations. Because the data and the information used to generate successful models throughout the hierarchical operations are made available through the accumulation concept that is adopted, this sub-component is capable of conducting the initial optimisations before it decides to loop back and send the model to a previous level for major changes. The algorithm of this subcomponent is designed to refer to the algorithms of the generative transformational and the evolutionary genetic subcomponents of the previous levels.

If a generated model needs major optimisation during the permutation or the propagation of new design solutions, the process of resolving the problem occurs by looping back to the original generation of the object in a

previous level or by referring back to the priority settings. Understandably, some problems or conflicts might not be resolved by the system and thus the solution would be deleted or the designer could be consulted.

5.1.5 Interface

Although the HEAD system is computer-based, the human designer is still a key part of the design process. The design problem is, at its most basic level, the requirements of the building and the constraints within which it has to be built. The design requirements and constraints and thus the design problem, (as discussed in Section 4.4.1) are found to be dynamic and changing throughout the problem-solving activity. As the design process progresses, so the design problem develops as the designer learns more about, and responds to, the changing needs and limitations of the design he is facing. As the designer begins to develop a design solution, more often than not, he will become aware of more or changing constraints. Thus, the design problem, whether it is the original problem contained in the design brief presented by the client or the design problem, which ultimately develops through the design process, is, by its very nature, subjective. The client, the user and the designer will all have subjective views and the interpretation of the design objectives and the design constraints. If the complexities of the design process are fully considered, it becomes clear that computer-aided design in general, and in the context of this research the HEAD System in particular, is and indeed should, *aid* the designer rather than try to replace him.

With regard to the HEAD system, abstracting the design concept into a design schema that is fed into a computer, would ideally involve producing lists of the relevant external and internal design constraints. The lists of the internal constraints will include a list of the required rooms with full descriptions of their spatial and functional requirements and their relationships with one another. Lists of the external design constraints would include lists of the site data that will be used for setting up the regulating elements that will govern the space layout planning and the massing generation of the design model. Obviously, all these activities

require a great deal of interaction between the designer and the HEAD system.

An effective, efficient and user-friendly interface between the computer and the human designer is, therefore, vital. A user-interface will enable the designer to enter relevant information into the system in addition to enabling him to observe and monitor (on both a visual/graphical and analytical level) the generation and evolution of the design solutions. When instances occur where the designer's input is required, the interface will provide a way for the designer to enter additional data to direct the design process in a way that reflects his judgments and preferences. In addition, the user interface allows the designer to interact with the HEAD system during the generation and optimisation processes, which is invaluable because, when conflicts occur, the designer's input is required before the system can promote a design solution to the next level. At the end of each level of the Synthesis Algorithms component, the generated solution is graphically represented for the designer to enter his feed back and subjective judgement. The designer is thus able to make particular changes to the results either by modifying the inputted data, which the HEAD system used in the first place, or imposing a certain direction for the system to take during the permutations and the propagation processes.

5.2 Illustrative paper-based simulation: Mosque

5.2.1 Bases of selecting the Mosque

As discussed in a previous chapter, (4), the design constraints can be categorised in many different ways. Categorisation is not clear-cut and this ambiguity of the design constraints contributes to a large extent towards the architectural design problems being ill-defined and, in many instances, dynamic. Some architects see the design task in terms of a process of identifying and satisfying design constraints (Lawson, 2006; Rowe, 1998). From the point of view of many architects, the uniqueness of each building stems from the various sources of the specification of the design problem that form the basis from which the architect extracts what he considers to be the design constraints. Design constraints, such as the building type, functional requirements, environmental factors, site condition, surroundings and rules and regulations, are all critical factors. Multiple constraints will exert a significant influence on a building design process and outcome. The design constraints often conflict with each other; however, usually one of the design constraints will stand out as the main influencing factor in the design process and solution. This does not in any way suggest that only the main constraint influences the design.

Lawson (2006) argues that it is not clear whether any particular constraints should be tackled first, whether any constraints are critical in determining the design and its success or indeed, whether different designers focus more on different types of constraints. In some cases, one major design constraint overrides other design constraints and determines the design success. For instance, the 'Right to Light' is an English law that gives the owner of a building with windows a right to preserve a suitable level of illumination based on an ancient light law (Kerr, 1865). Since the early twentieth century, when a new buildings was erected for planning and for legal cases where the 'Right of Light' is brought to bear, Waldram's procedure has been accepted as a means of establishing that there will be adequate potential for daylight for both the new building and for the

existing surrounding buildings (Chynoweth, 2004, 2005, 2009; Peter & Ian, 2007). Despite the long debate about the validity of Waldram's measures (Chynoweth, 2004, 2005, 2009; Peter & Ian, 2007), the fact remains that a single constraint is able to shift the design process and the design solution in a certain direction. Thus, the designer can very often find himself forced to focus more on the most decisive constraint.

There are many examples of dominant constraints in buildings design. Occupant circulations in airports, hospitals and museums are usually one of the most dominant of the design's constraints that arise from the problem specification. Security measures are a key determinant in the design success of high-level governmental buildings, such as embassies and military complexes. Building laws and regulations, such as set backs, building heights and the allowable building area, are also found to play, in some cases, a significantly important role and are often prioritised by architects (depending on the particular design problem at hand) to be the most dominant of the constraints of a design problem.

Needless to say, when the designer is faced with a design problem that includes the dominant constraint(s) (as interpreted by the individual designer), it does not mean that other types of constraints do not act as influential factors in the measuring of the design success. Also, it is not unusual for a designer to face conflicting constraints while tacking a design problem originally specified in the design problem or generated during the search for a design solution. As discussed in the previous chapter (Design Method), conflicting design constraints can be dealt with by prioritising them by degree of importance.

Within the context of mosque design, mosques have more defined design problems because specific planning and design criteria arise from a set of explicit religious requirements. For instance, the orientation towards Makkah for the prayer hall and the module of the prayer mat within the prayer hall are found to be the two overriding design constraints. The courtyard and ancillary spaces can be oriented in whichever direction is

appropriate. These distinguishing features provide a greater restraint over the design variants and thus a more restricted design space and generation of solutions. Because the design problem faced when designing a mosque offers a more defined and controlled design environment than in most other design situations, mosque design was chosen for the research illustrative paper-based simulation. The controlled design problem, design space and generated solutions also provide a reference for the functional evaluation of the produced design solution. In addition, having almost all the design criteria predetermined, mosque design is believed to present a meaningful challenge to the evolutionary algorithm in the exploration of the solution space and the generation of the most suitable design solution.

Another reason for choosing the Mosque for the illustrative paper-based simulation arises from the author's practical experience at the General Directorate of Military Works (GDMW), the building procurement arm of the Saudi Arabian Armed Forces. The GDMW projects are managed completely by one body, from inception to eventual maintenance. The standardization of the repeated projects has been a controversial issue among GDMW's decision-makers. One school of thought argues that using a standard design saves on design time in the repeated, less important projects and thus allows for a more concentrated effort on the selected, more important projects. The other school of thought believes that each building is unique, and should thus be designed in response to its needs and the environment.

GDMW standardized projects have been developed over time and have a number of uses. All their related data is, in theory, gathered and processed, which provides suitable testing material. Selecting one particular standardized project, such as mosques, will allow control over design variants to regenerate different design solutions by only applying changes to the design schema or by a change in the environment. The proposed design method is expected to allow the standardization of the design requirements and elements in addition to the design schema. Each time the

same building is needed in a different location, a new and unique design evolves in response to its living environment.

The proposed design method support a design process that could be used for any design, but that could also be standardised to a certain degree for it to be used to develop particular building types. Even a standardised process would produce design solutions that would vary according to the specific environment of the project. The proposed design method is thus believed to have the potential for offering a middle ground between the proponents of standardisation and the proponents of the architectural principle that considers each project as unique.

The design schema within the proposed design method is conceived on a task-specific basis; that is to say, for the illustrative paper-based simulation of the HEAD system, it is specifically designed for a mosque design project. Notwithstanding the very focused design schema described for the illustrative paper-based simulation, it is believed that the overall approach of the HEAD system can be applied to any type of building, because individual designers are expected to produce relevant task-specific design schemas in response to the specific design problems they face.

5.2.2 System operation: Mosque design

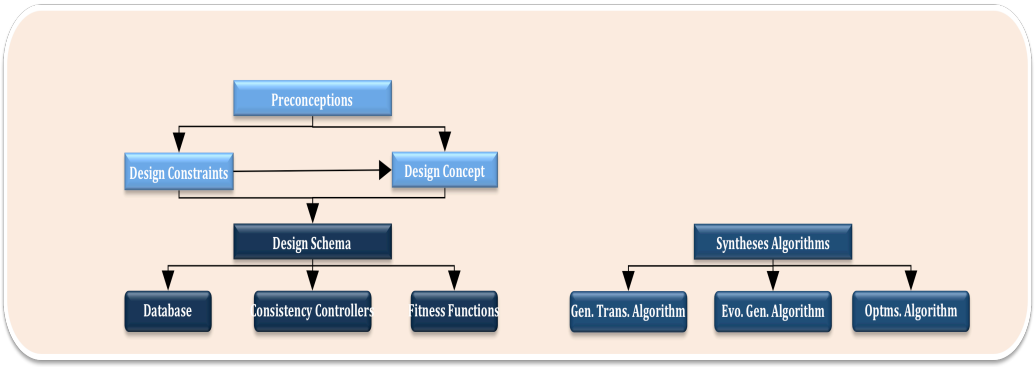


Figure 5-7 System main components: Mosque design

The followings sections describe a paper-based simulation to illustrate how the system is expected to operate through the Mosque project design process.

5.2.2.1 Room level

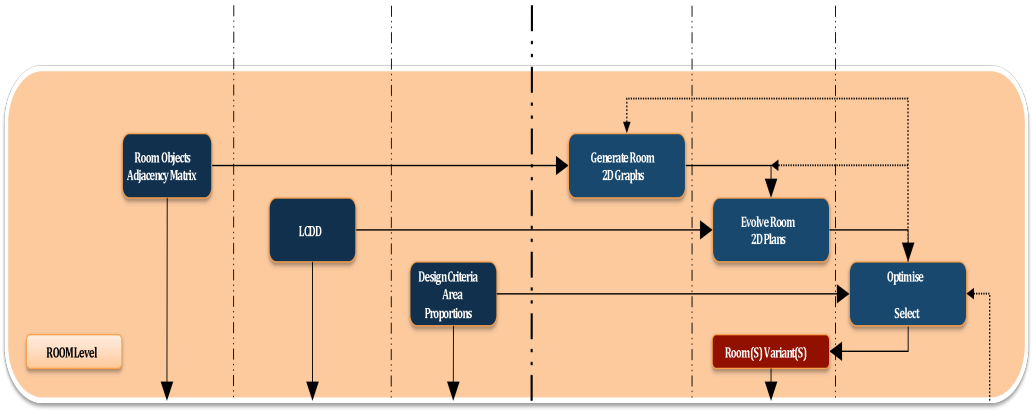


Figure 5-8 Mosque's room level

5.2.2.1.1 Room level database

5.2.2.1.1.1 Room objects

Each room of the mosque is decomposed into its constituent components and objects. Based on architectural design standards, each object is analysed according to the three key categories of spatial artefacts, 'object space', 'operational space' and 'functional space', identified by Bhatt et al. (2009; 2010), and previously discussed in Section (4.6.2.1). This form of the representation of the objects and their related spatial artefacts translates the spatial requirements into a modular basis that enables the generative algorithm to produce different variants for each space. Each variant is assessed, based on the assigned fitness functions; those variants scoring the highest are fed into the layout planning stage. If it is found to be necessary in a later phase, looping back to this stage may also optimise a selected variant.

A review of the GDMW's standard Mosque designs was conducted to check that the dimensions and area for each object fall in line with the general architectural design standards (De Chiara & Callender, 1980) and the Mosque design criteria, as discussed previously. The elements and objects of each space were abstracted and analysed to indicate (A) the object space, (B) the operational space and (C) the functional space. It is worth mentioning here that the functional space (C) of one object might overlap with the functional space of an adjacent object.

To demonstrate the decomposition of the space into its constituent elements and objects, the prayer hall and the ablutions area were selected. The following diagrams show the essential elements and objects of the prayer hall and the ablution area and toilets. (Figure 5-9 Prayer Mat), (Figure 5-10 Entrance door) with the transition area, (Figure 5-11 Typical formation of the Imam's entrance, Mihrab & Minbar), (Figure 5-12 Spatial analysis of the Imam's entrance, Mihrab & Minbar), (Figure 5-13 Structural Column), (Figure 5-14 Walls), (Figure 5-15 Bench), (Figure 5-16 Shoe rack), (Figure 5-17 Bench & shoes rack configuration 1) and (Figure 5-18 Bench & shoes rack configuration 2). Diagrams showing the elements and objects of the ablutions area and the toilets are; (Figure 5-19 Section detail of the ablution area), (Figure 5-20 Spatial analysis of the ablutions area), (Figure 5-21 Spatial analysis of the toilets) and (Figure 5-22 Spatial analysis of wash areas).

Prayer hall

Prayer mat

The prayer mat, Figure 5-9 Prayer Mat, module is confined to the adequate space needed for the prayer to perform the ritual of prayer. Note that, here, both the operational (B) and the functional spaces (C) fall within (a) the object space. The reason for this is because the modules will organise a grid configuration of rows and columns where they are directly adjacent to each other.

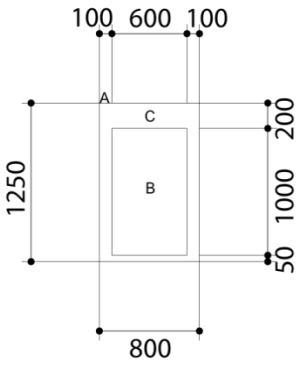
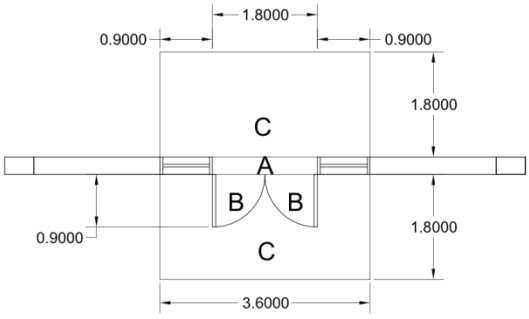


Figure 5-9 Prayer Mat

Entrance door

The functional space (C) of the entrance door, Figure 5-10 Entrance door is the transition area of the Mosque. The area needed for the transition area is usually determined by the number of pedestrians passing the door within a certain range of time. The main prayer hall has a



total of seven doors distributed between three entrances, serving a total of 1500 prayers. The evacuation time after the prayer ends is the critical time for the calculation of the required space by the sides of the doors, because the prayers arriving at the Mosque prior to the pray time have a wider time range between the call to prayer and the start of the prayer. When the prayer ends, the area of the prayer hall, which is open to the transition area by the door, could accommodate the crowd waiting to pass through the door bottleneck.

Figure 5-10 Entrance door

The Imam entrance, Mihrab, and the Minbar

These essential elements of the mosque have a spatial relationship and configuration. The Imam mainly uses these three elements in the same sequences of activities that he performs during the prayers (Friday prayers in particular). The integrated functions determine the spatial configuration between these three elements, Figure 5-11 Typical formation of the Imam's entrance, Mihrab & Minbar. The Imam approaches the Minbar for the Friday speech, whether through his door or from inside the prayer hall. When the speech is over, the Imam steps down from the Minbar to lead the group prayer at the Mihrab.

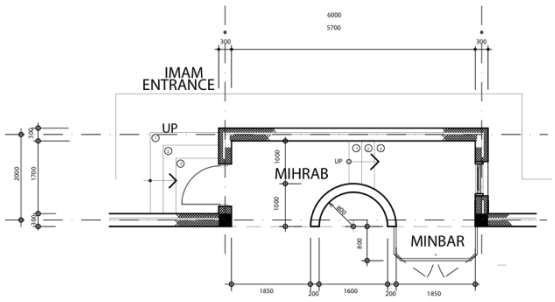


Figure 5-11 Typical formation of the Imam's entrance, Mihrab & Minbar

Figure 5-12 Spatial analysis of the Imam's entrance, Mihrab & Minbar. The Imam entrance (A1), Mihrab (A2), and the Minbar (A3) share adjacent and/or overlapping functional spaces (C) as shown in the diagram. The Mihrab accommodates half of the Imam's prayer mat and the other half lies in front of the first row behind him and in-line with the edge of the Mihrab edge, which is

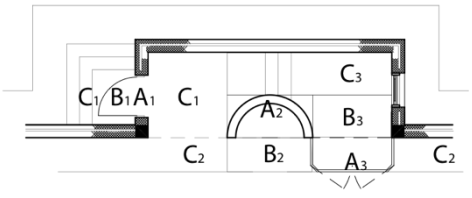


Figure 5-12 Spatial analysis of the Imam's entrance, Mihrab & Minbar

penetrating out of the Qiblah wall. The side area in front of and together with the first row, which are considered the Mihrab's functional spaces (C2), provide comfort for the first row of prayers and also provide a suitable location for the Qur'an stands.

Structural column

One of the constraints on the prayer hall designs is that the rows should not be divided and therefore the columns should not be placed in the middle of the row. Taking this into consideration, the two sides of the column, Figure 5-13 Structural Column, are considered as their operational space and 100 mm to both the top and bottom sides are considered as the functional space for the comfort of the worshippers.

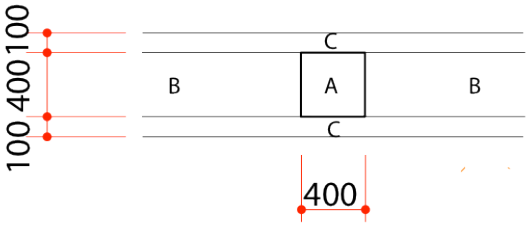


Figure 5-13 Structural Column

Exterior wall

Generally, the room boundaries are built with walls, both exterior and interior according to its location and neighbouring spaces. In the case of the prayer hall, it is one large space surrounded by exterior wall on all four sides.

The object pace (A) and the functional space (B) of the wall are within the wall itself. The operational space (C) is just a small distance right in front of the wall Figure 5-14 Walls.

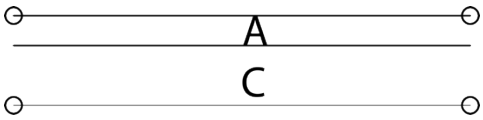


Figure 5-14 Walls

Bench

Benches are used to accommodate those who need to sit down while taking off or putting on their shoes. The functional space (B) surrounds the object space (A) and surrounded by the operational space Figure 5-15 Bench.

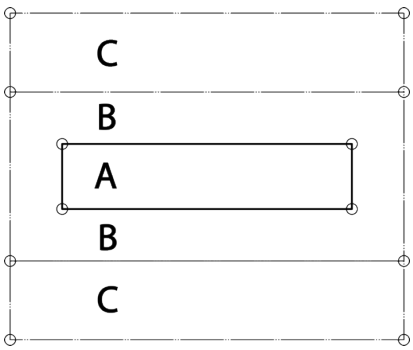


Figure 5-15 Bench

Shoes rack

The diagram in Figure 5-16 shows a mirrored shoe rack configuration. The functional space (B) right in front of the shoe rack is to allow ease of storing and retrieving of the shoes. The operational space (C) provides ease of access and passage.

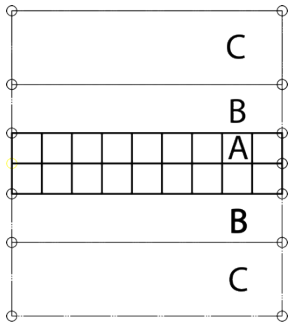


Figure 5-16 Shoe rack

Bench and shoe rack configuration

Figure 5-18 and Figure 5-18 show two different configurations of the shoe rack and the bench. As mentioned earlier, it is possible for the operational space of two objects to overlap as the case in Figure 5-17 where the two objects are facing each other. The other case, presented here in Figure 5-18 for a side-by-side configuration. According to the required number of benches and shoe racks, the system will explore all the possible arrangements in relationship to their adjacency requirement with other objects.

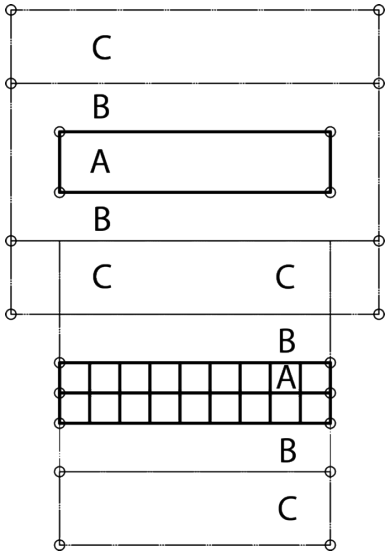


Figure 5-17 Bench & shoes rack configuration 1

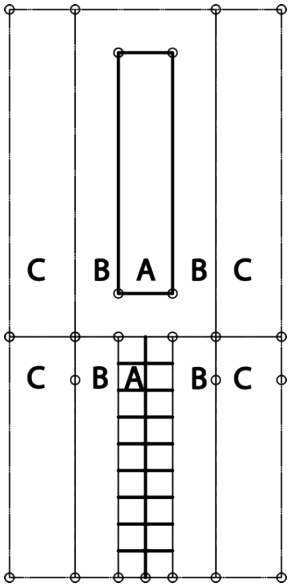


Figure 5-18 Bench & shoes rack configuration 2

Ablutions and toilets

Ablutions

The ablutions are is typically a floor level washing facility, Figure 5-19 Section detail of the ablution area, where the faucet is placed in front of a person sitting. The reason for such a design is mainly to accommodate the part of the ablution ritual that involves washing of the feet.

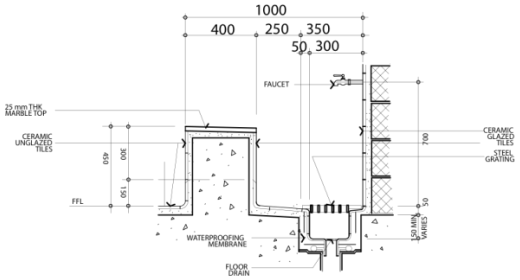


Figure 5-19 Section detail of the ablution area

The seat can be more clearly visualised using **Error! Reference source not found.** where the object space (A) is surrounded by the operational space (B) where the actual ritual is conducted. The functional space (C) is right behind the seat allowing for easy access.

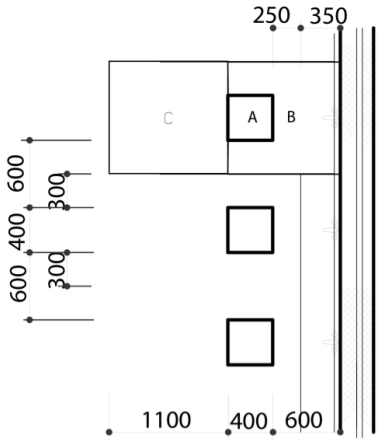


Figure 5-20 Spatial analysis of the ablutions area

Toilets

The toilet cubical, Figure 5-21 Spatial analysis of the toilets, is considered as an object itself thus, its door is the object space (A) and space inside the cubical is the operational space (B), while the access area in front of the door is the functional space (C).

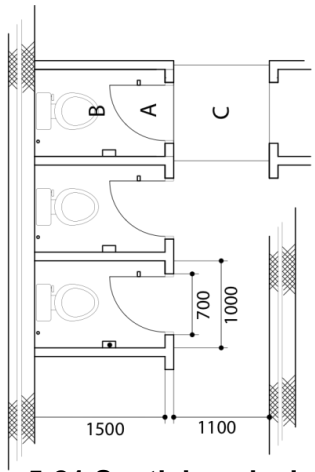


Figure 5-21 Spatial analysis of the toilets

Basins

The basin, Figure 5-22 Spatial analysis of wash areas, occupies the object space (A) while the area in front of it is the operational space (B) for the user and right next to this is the functional space (C) for access.

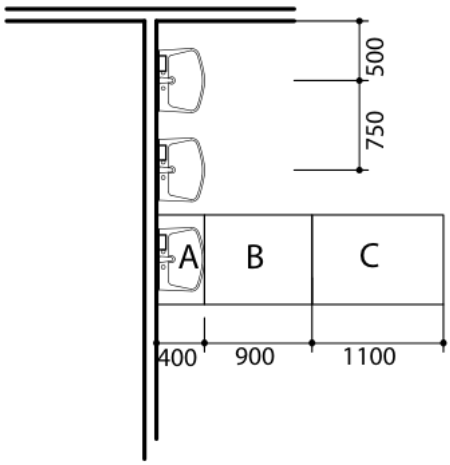


Figure 5-22 Spatial analysis of wash areas

5.2.2.1.1.2 Objects adjacency relations

Prayer hall adjacency relations graphs

Objects in each room of the building were determined, based on the room functional requirements and the activity it is intended to host.

The adjacency matrix of the relationships between the room objects provides alternatives and therefore a number of different graphs can be generated for each room.

Figure 5-23, Figure 5-24 and Figure 5-25 illustrate three different alternative graphs generated out of the prayer hall objects adjacency relations' matrix.

Key;

(1) Worship area
(2) The Imam entrance, Mihrab, and the Minbar
(3) Entrance
(4) Shoe racks

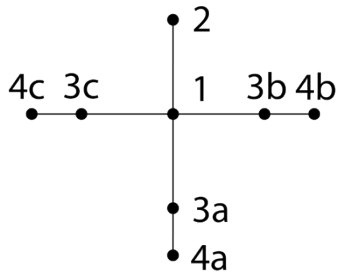


Figure 5-23 Prayer hall adjacency relations, graph 1

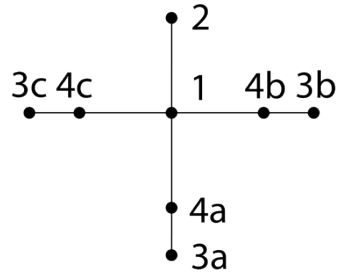


Figure 5-24 Prayer hall adjacency relations, graph 2

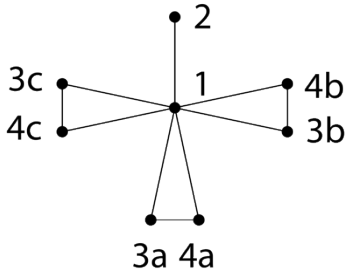


Figure 5-25 Prayer hall adjacency relations, graph 3

Ablution and toilet area adjacency relations graphs

Figure 5-26, Figure 5-27, Figure 5-28 and Figure 5-29 illustrate three different alternative graphs generated out of the prayer hall objects adjacency relations matrix.

Key;

(1) Entrance
(2) Ablutions area
(3) Basins area
(4) Toilets area

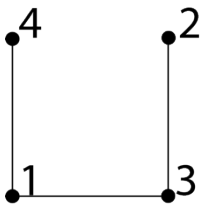


Figure 5-26 Ablution & toilet area adjacency relations, graph 1

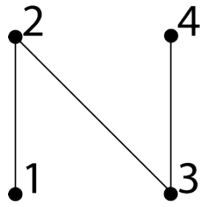


Figure 5-27 Ablution & toilet area adjacency relations, graph 2

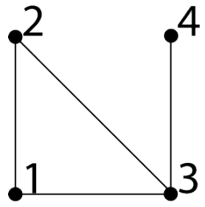


Figure 5-28 Ablution & toilet area adjacency relations, graph 3

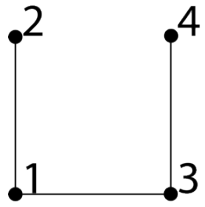


Figure 5-29 Ablution & toilet area adjacency relations, graph 4

5.2.2.1.1.3 2D LCDD module

To demonstrate how the LCDD concept can be applied in various ways, the overlapping shifting grids of the site of a mosque project can be used. The City Grid (CG), Qiblah Direction (QD) and the True North (TN) (solar orientation) coordinates are used to create a three-layer weave with a simple fixed distance array of the x and y of each coordinate. The LCDD can also be chosen to represent a functional meaning such as the prayer matt, in the case of the Mosque.

City Grid (CG), Qiblah Direction (QD), True North (TN)

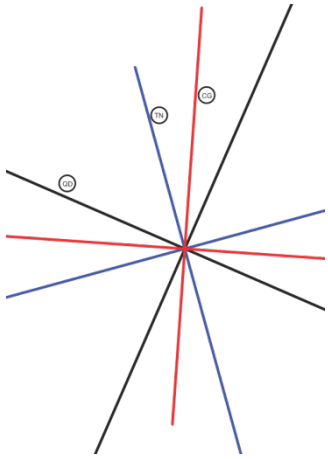


Figure 5-30 site coordinates

Generate Overlapping shifting grids;
Array x, y of CG, 1m
Array x, y of QD, 1m
Array x, y of TN, 1m

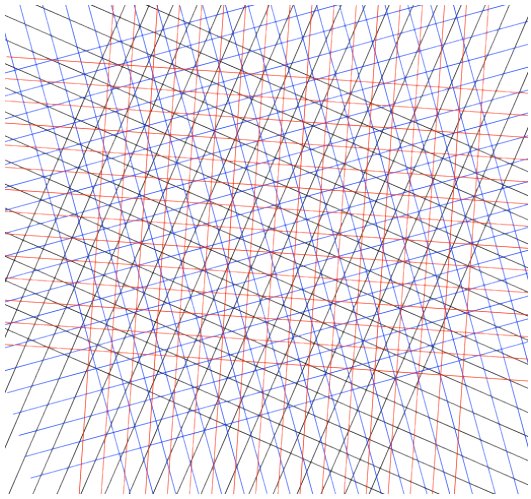


Figure 5-31 generated overlapping grids

Create a random shape of n_intersection points
(n=6, designer input!)
CG_line intersects with QD_line and TN_line
QD_line intersects with CG_line and TN_line
TN_line intersects with CG_line and QD_line

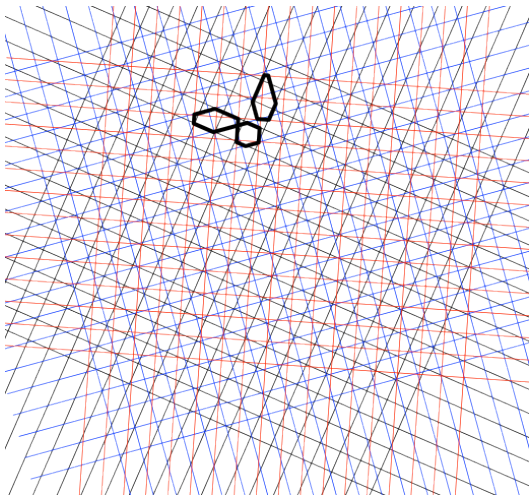


Figure 5-32 search LCDD

Plot LCDD shape(s)

Figure 5-33 LCDD 1

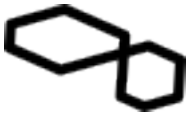


Figure 5-34 LCDD 2

Array LCDD shape on matching intersecting points over the entire three overlapping grids and within the buildable area

• If patterns match
yes = go to next step
no = go back, try next intersection point
• Plot pattern

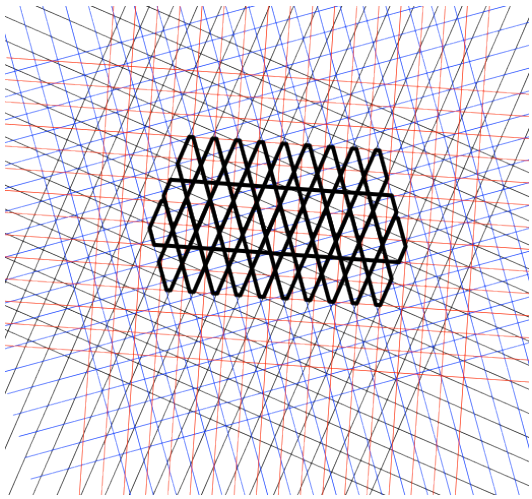


Figure 5-35 LCDD 1 pattern

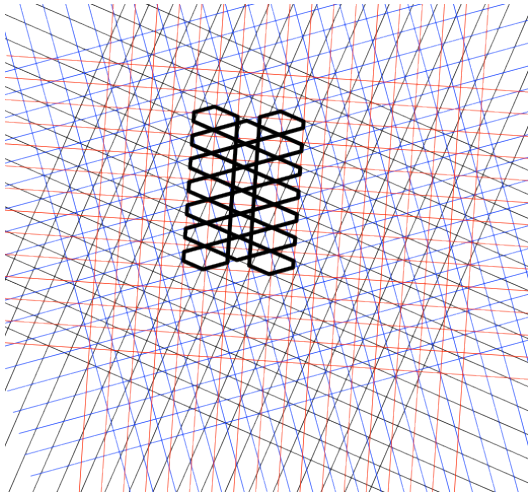


Figure 5-36 LCDD 2 pattern

The prayer mat is a simple rectangle representing the prayer's spatial requirement for conducting the ritual. Figure 5-37 the prayer's matt as an LCDD, illustrates the utilisation of the prayer mat as a room object and an LCDD to generate uniformed spaces.

In this thesis, and for the purpose of simplicity of illustration, the prayer mat will be used as the LCDD.

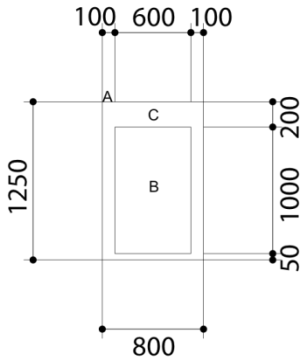


Figure 5-37 the prayer's matt as an LCDD

5.2.2.1.2 Room level consistency controller

5.2.2.1.2.1 LCDD

The generation of rooms can be regulated by the use of a two-dimensional LCDD module that will not produce rooms with arbitrary dimensions and will thus facilitate the placing of adjacent rooms together. The use of the 2D LCDD for generating rooms lays the foundations for the layout planning and structural component allocation that will be undertaken in later stages to generate the form of the building.

5.2.2.1.3 Room level fitness functions

- Architectural design standards
- Design criteria

- Area
- Proportion

5.2.2.1.4 Room level generative transformational algorithm

A two-dimensional dual graph of the relevant geometrical properties of each object can be used to depict the relationships between the different room objects. The designer will select the planner module (LCDD) that can then be multiplied and arranged in a variety of ways to accommodate the dual graph representation of the room objects. Because the dual graph incorporates not only the room objects but also the relationships between the objects, the room's functional requirement can be met.

The adjacency requirements of the objects are translated into a graph by the generative transformational algorithm at the room level as shown in Figure 5-23, Figure 5-24 and Figure 5-25. The graph is subsequently developed into a dual graph with the dimensions of each object being taken into account. Figure 5-38 Prayer hall adjacency relations, dual graph 1, Figure 5-39 Prayer hall adjacency relations, dual graph 2 and Figure 5-40 Prayer hall adjacency relations, dual graph 3 show the translation of two examples of the prayer hall graphs into rectangular dual graphs.

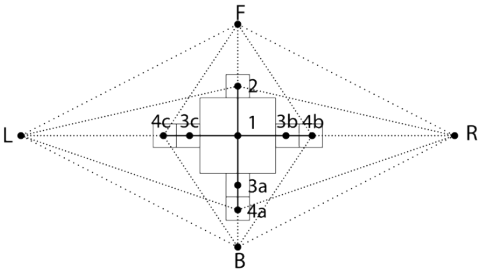


Figure 5-38 Prayer hall adjacency relations, dual graph 1

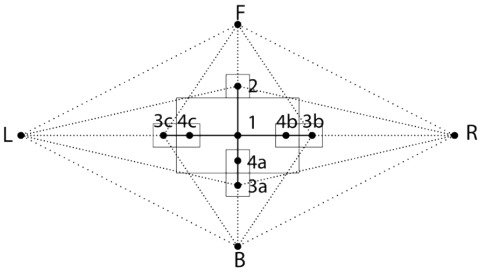


Figure 5-39 Prayer hall adjacency relations, dual graph 2

Key;

(1) Worship area
(2) The Imam entrance, Mihrab, and the Minbar
(3) Entrance
(4) Shoe racks

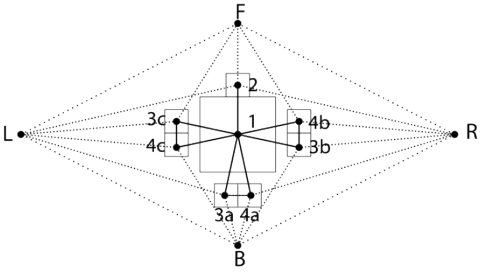


Figure 5-40 Prayer hall adjacency relations, dual graph 3

5.2.2.1.5 Room level evolutionary genetic algorithm

Several different adjacency relations' graphs and rectangular dual graphs can be used to represent the adjacency requirement between objects within the room. The dual graphs used can then be used to represent various arrangements of planner modules.

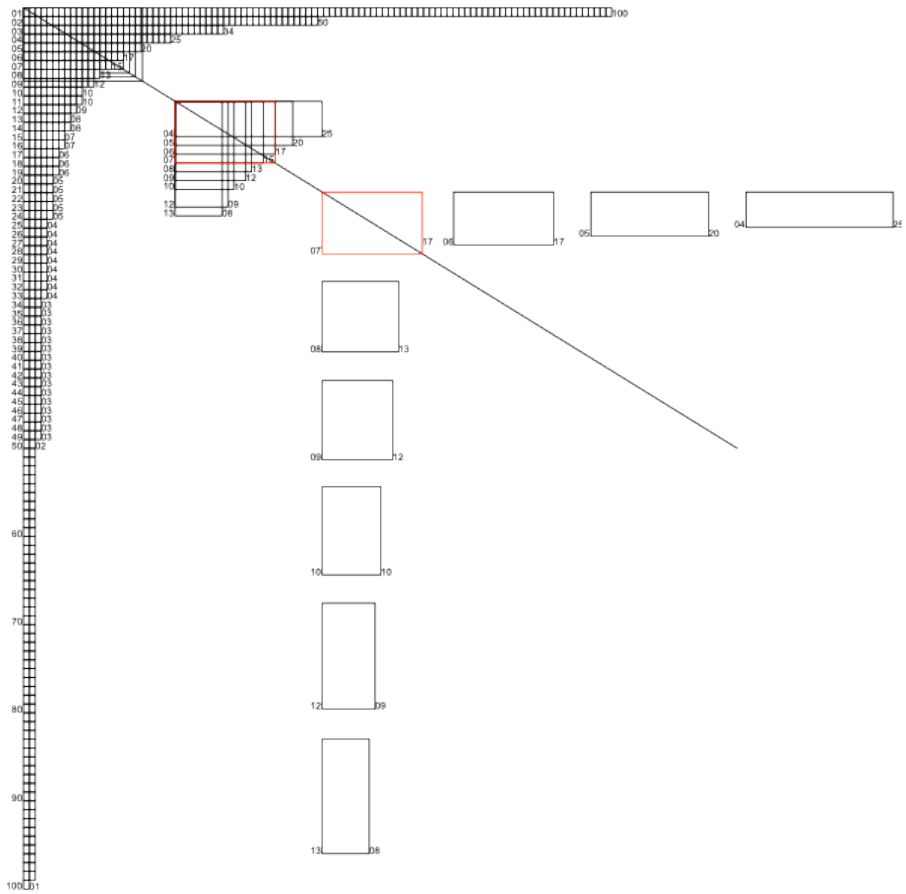


Figure 5-41 Prayer Hall variants

Figure 5-41 Prayer Hall variants, shows the different possible variants generated according to the required capacity. The rectangular shaped prayer halls are found to be the most appropriate, because rows of equal lengths can be achieved. The ideal situation is to have the longer side of the rectangle of the prayer hall facing the Qiblah direction towards the Kaa'ba, so that the longest possible rows of prayers can be accommodated. The system generates all the possible variants and determines the few successful plans that are in the middle of the two extremes (one row or one column).

Figure 5-23, Figure 5-24 and Figure 5-25 shows several variations on setting the adjacencies of the Prayer Hall. The translated adjacency graph is then depicted in different dual graphs; Figure 5-38, Figure 5-39 and Figure 5-40.

The relationships between the dual graph of the room objects, and the arrangement of the planner modules hosting it, will provide the evolutionary algorithm with suitable representations to use during the genetic operations. What this means is that the dual graph of one parent can, for instance, be processed, through genetic operations, with the arrangement of the planner modules of the other parent.

Figure 5-43, Figure 5-43, Figure 5-44, Figure 5-45, Figure 5-46 and Figure 5-47 showing different prayer hall room variants which translate the room components adjacency relations' graphs and the dual graphs representing each one.

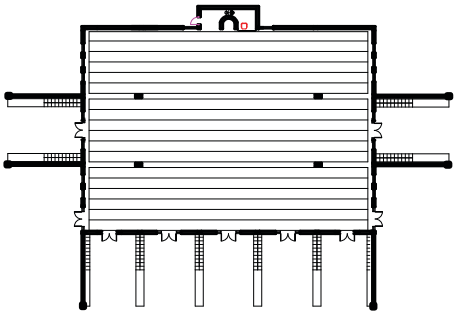


Figure 5-42 Prayer hall variant 1

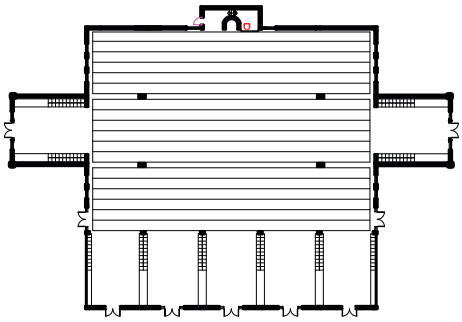


Figure 5-43 Prayer hall variant 2

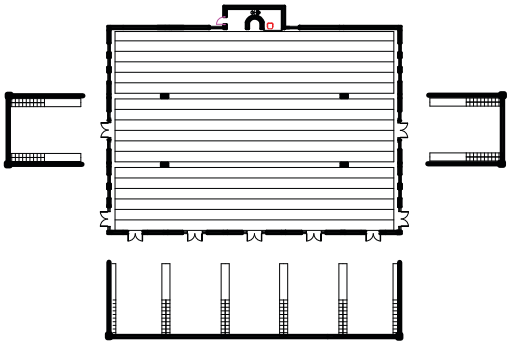


Figure 5-44 Prayer hall variant 3

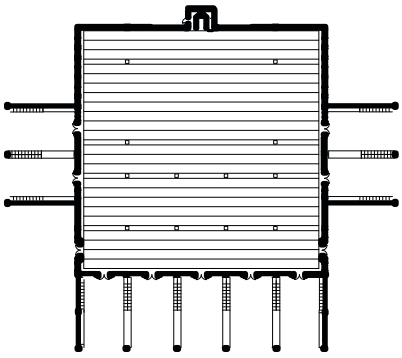


Figure 5-45 Prayer hall variant 4

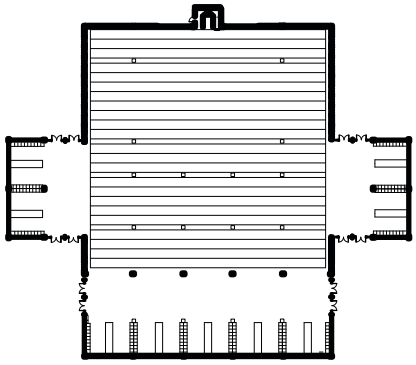


Figure 5-46 Prayer hall variant 5

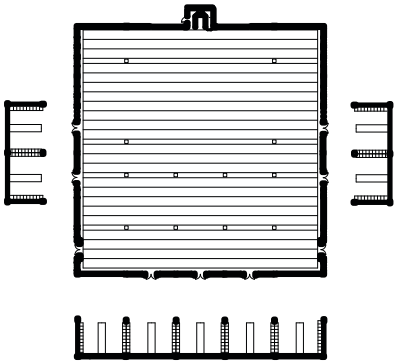


Figure 5-47 Prayer hall veriant 6

Similar to the prayer hall generated variants, Figure 5-49, Figure 5-49, Figure 5-50 and Figure 5-51 showing different variants of the Ablutions and toilets area which translate the room components adjacency relations' graphs and the dual graphs representing each one.

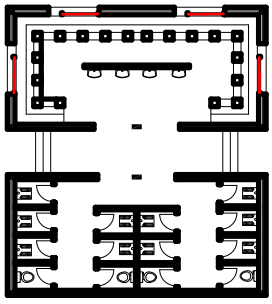


Figure 5-48 Ablutions and toilets variant 1

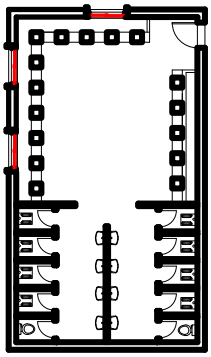


Figure 5-49 Ablutions and toilets variant 2

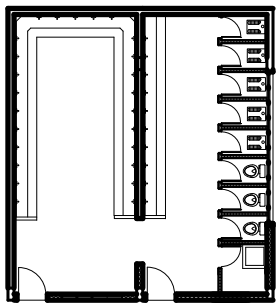


Figure 5-50 Ablutions and toilets variant 3

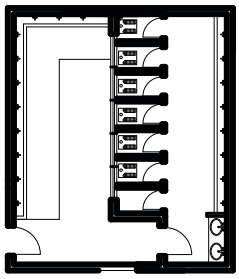


Figure 5-51 Ablutions and toilets variant 4

5.2.2.1.6 Room level analysis, evaluation and selection processes

The resulting generations that have evolved from the genetic operations can then be analysed and evaluated. Each generation will be assigned a score that indicates the fitness of that generation. The fitness functions used at the room level are the adjacency constraints, area, perimeter and proportion, in addition to the compliance with any specific spatial requirements.

Selection should be based on the highest fitness function score. The variants of each room selected are fed into the database of the layout level of the system. The successful room level variants are assigned a rank according to their score, and this rank is also fed into the layout level database and can be used for space layout planning.

5.2.2.2 Layout level

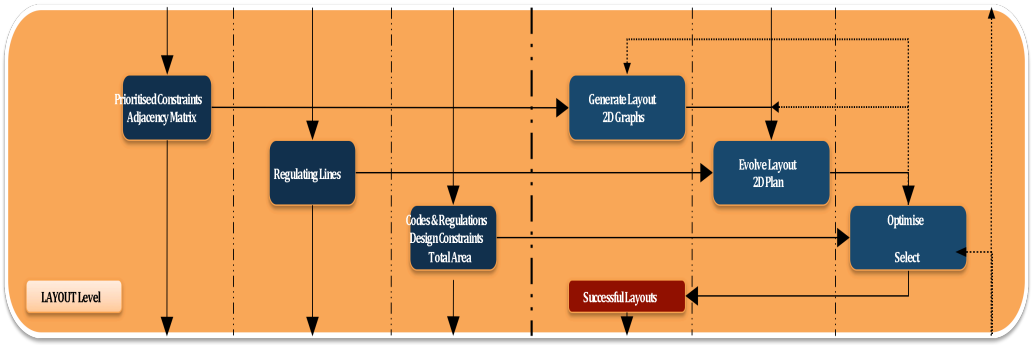


Figure 5-52 Mosque's layout level

5.2.2.2.1 Layout level database

5.2.2.2.1.1 Constraint prioritisation

Clearly, the orientation to Makkah is the overriding design constraint in the case of mosque design in general. Because the Earth is spherical, we can find the most direct orientation to Makkah from any point on Earth by using the general formula used to connect two points on a sphere. The direction between any two points on Earth (or any sphere) together is the shortest path that connects the points on the surface. It is generally accepted that the shortest path lies on an arc of a circle that passes through the two points and has its axis at the centre of the sphere (also known as a Great Circle).

To find the Qibla direction from any point on Earth, the one-line formula for a great circle can be used (Zarrabi-Zadeh, 2007–2010).

$$\alpha = \arctan [\sin(\lambda 2 - \lambda 1)/\cos \varphi 1 \tan \varphi 2 - \sin \varphi 1 \cos (\lambda 2 - \lambda 1)]$$

where $(\varphi 1, \lambda 1)$ is the latitude-longitude of your location, and $(\varphi 2, \lambda 2) = (21.25, 39.49)$ is the latitude-longitude of Kaa'ba. The returned value of α, specifies the angle of Qibla clockwise from true North.

The prayer's spatial requirements are represented by the prayer mat and govern the size of the worship areas according to the overall required capacity. Additionally, the order of the formation of the prayers during the prayer time in rows influences the shape of the worship areas. A

rectangular shape is thus found to be the most efficient plan for the prayer hall design.

The next most influential consideration in mosque design is the function requirement for the mosque to provide the users with a spiritual experience when conducting their rituals. If a mosque design cannot help to enhance the spiritual experience, the design's success will be called into question. The functionality of the design solution comes from the well-thought-out set-up of each room's spatial functional requirements and the consideration of the spatial relations and adjacency requirements. The achievement of a high performance for the building in response to its environment is an added measure for the functionality and the quality of the design solution.

Apart from the spatial and functional requirements, one religious aspect poses an important constraint on the design of the toilets. As stated earlier, the orientation of the toilet seats should not be in line with the Qiblah direction. Additionally, it is preferably that the ablutions and toilet areas be located at the back of the Mosque.

5.2.2.2.1.2 Spatial adjacency requirements

The weighted adjacency matrix of the adjacency rules and requirements for the design is based on the spatial and functional relations between the building's spaces. The adjacency matrix provides multiple scenarios that will result in the generation of a variety of overall building configurations. As in the case of the room objects, the use of a weighted adjacency matrix for the building layout provides alternatives for the designer and therefore can be used to generate different graphs.

A matrix was developed for the mosque's spatial and functional relations between the building spaces, Section (7.8).

5.2.2.2.2 Layout level consistency controller

5.2.2.2.2.1 Regulating lines

In urban design, SLOAP (Tanghe, Vlaeminck, Berghoef & Southam, 1984) is a pejorative acronym for 'Space Left Over After Planning' refers to an irregular, inaccessible and usually small plot of land that is found to be un-usable. Ideally, this wasted SLOAP should be allocated for a suitable use. However due to boundary lines and accessibility issues and the like, this is not always possible, even in the case of utilising the SLOAP as an open space. Site topographical constraints and the angle of the grid system, as well as roads, are common generators of such residual spaces after planning.

Very often designers are faced with having to deal with multiple orientation issue when designing buildings that can lead to SLOAP within the building itself. The overlapping shifting grid presented in this thesis, using the three coordinates of true north, city grid and direction provides a means for avoiding this type of *internal* SLOAP when designing a building under such site constraints.

The technique presented in this thesis of generating an ordered grid system for design, using regulating lines descending from the site properties, offers a useful and effective means of tackling a design problem of multiple orientation constraints without creating wasted, left-over spaces both inside and outside the building.

With the aim of developing a design that is cohesive within its environment, foreign solutions drawn from unrelated sources should not be imposed on the design. During the site planning and analysis process, the designer identifies aspects such as location, neighbourhood context, size and zoning, legal elements, natural physical features, man-made features, circulation, utilities, sensory, human and cultural, and climate components. The physical, biological and cultural attributes of the site are constraints to the design and also steer the designer in particular directions for that particular design.

The designer may invent his/her own grid system or parti. The following diagrams show a simple generation method of a grid system with respect to the site properties that could be used as the design's regulating lines. The method is based on the three main coordinate systems of the site, City Grid (CG), Qiblah Direction (QD) and True North (TN). Start by placing the x-axis and the y-axis of each coordinate at each corner of the site; some will eventually intersect. At each intersecting point of the x and y axes of all coordinates, the other x and y axes of each coordinate are placed. As a result, more intersecting points will appear after each iteration. A test software was developed to demonstrate this concept (see Section 7.5). The following Figure 5-53 to Figure 5-60 generated by the software developed based on this simple algorithm.

City Grid (CG), Qiblah Direction (QD), True North (TN)

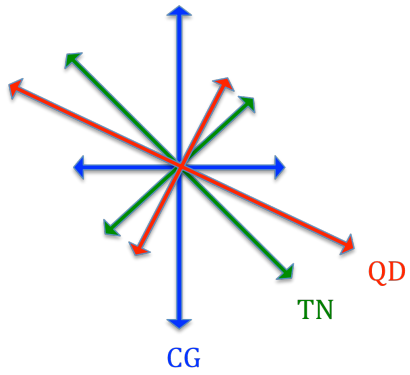


Figure 5-53 Site Coordinates

Place CG, QD and TN coordinates axis in each corner of the site

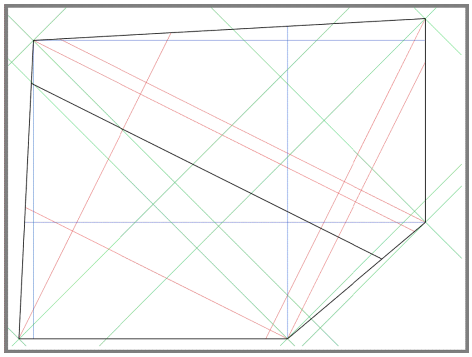


Figure 5-54 placing coordinate axis

(x) coordinate of one grid intersects with (y) coordinate of the other grid

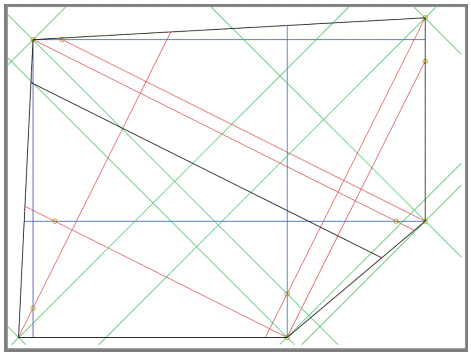


Figure 5-55 1st iteration; intersecting points

Place the other (x) and (y) coordinates of each grid

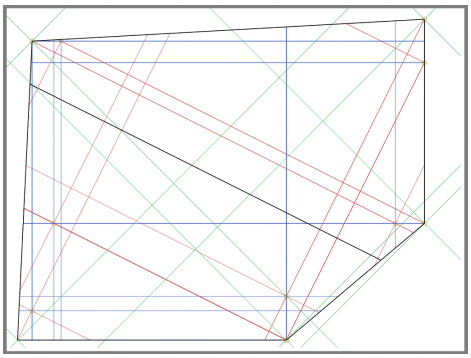


Figure 5-56 1st iteration; placing coordinate axis

(x) coordinate of one grid intersects with (y) coordinate of the other grid

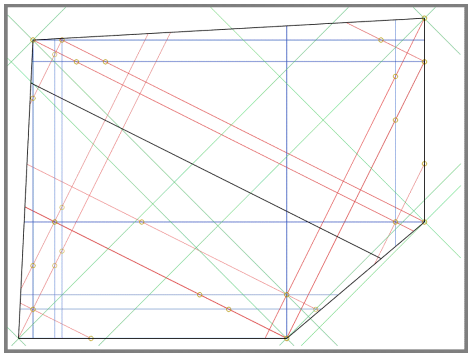


Figure 5-57 2nd iteration; intersecting points

Place the other (x) and (y) coordinates of each grid

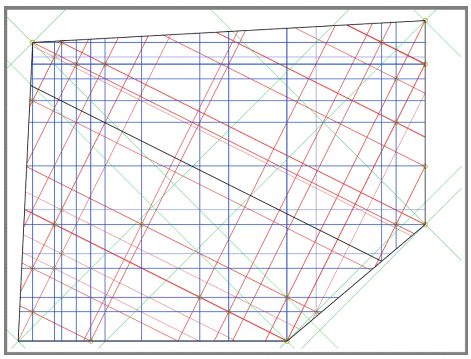


Figure 5-58 2nd iteration; placing coordinate axis

(x) coordinate of one grid intersects with (y) coordinate of the other grid

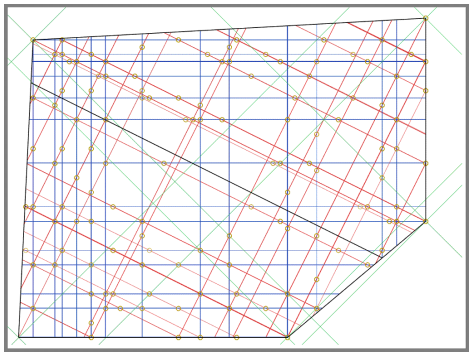


Figure 5-59 3rd iteration; intersecting points

Place the other (x) and (y) coordinates of each grid

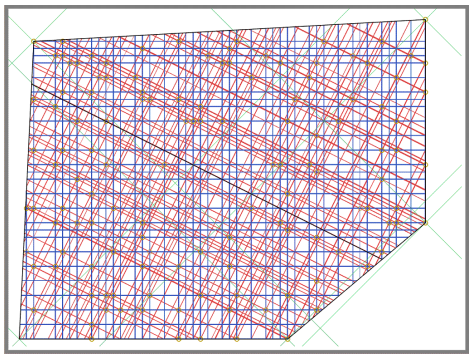


Figure 5-60 3rd iteration; placing coordinate axis

5.2.2.2.3 Layout level fitness functions

- Building codes
- Design constraints
- Total area

5.2.2.2.4 Layout level generative transformational algorithm

The generative transformational algorithm translates the adjacency matrix to the graphs starting with the highest adjacency preference. Then the algorithm starts to search for the right combination of rooms out of the previously-produced variants for each room. Each node of the graph is substituted with a particular room variant, with an integer number referring to its performance score, which best fits with the adjacent rooms. The result is a dual graph generated for each successful combination.

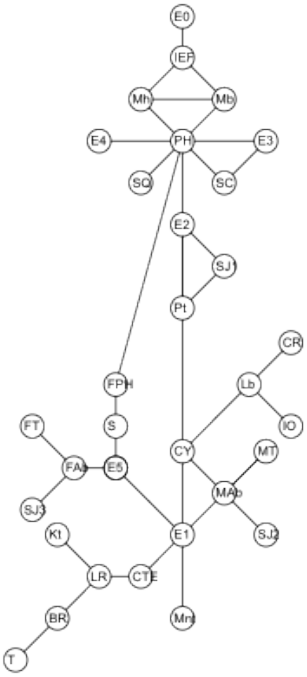


Figure 5-61 Parent A

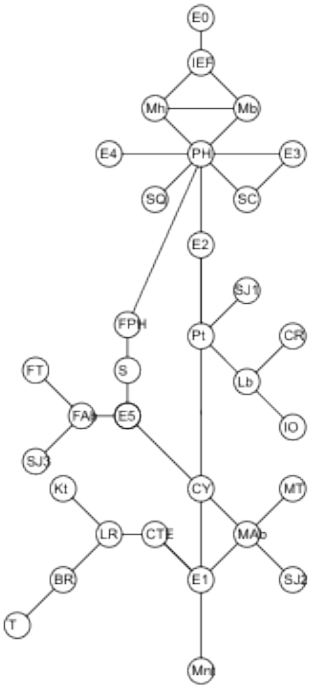


Figure 5-62 Parent B

Parents
A graph representation generated with successful configuration of room varaints that assigned with specific integer numbers.

5.2.2.2.5 Layout level evolutionary genetic algorithm

The various configurations of the building layout, which result from the generative transformational algorithm sub-component, can be graphically represented in dual graphs. These dual graphs should have integer numbers at nodes indicating the selected room variant embedded within them and can thus be used as the first or initial generation that would be feed into the evolutionary algorithm to start the propagation process. The genetic operations of the evolutionary algorithm can take place between either the dual graphs (representing the adjacency requirements) of both parents or the dual graph of one parent and the room variants of another parent or, indeed, between the room variants of both parents. The process of the propagation from the parents, guided by the regulating lines forming a grid system, can be used for reference during the search for the most suitable room dimensions and shapes among the generated variants.

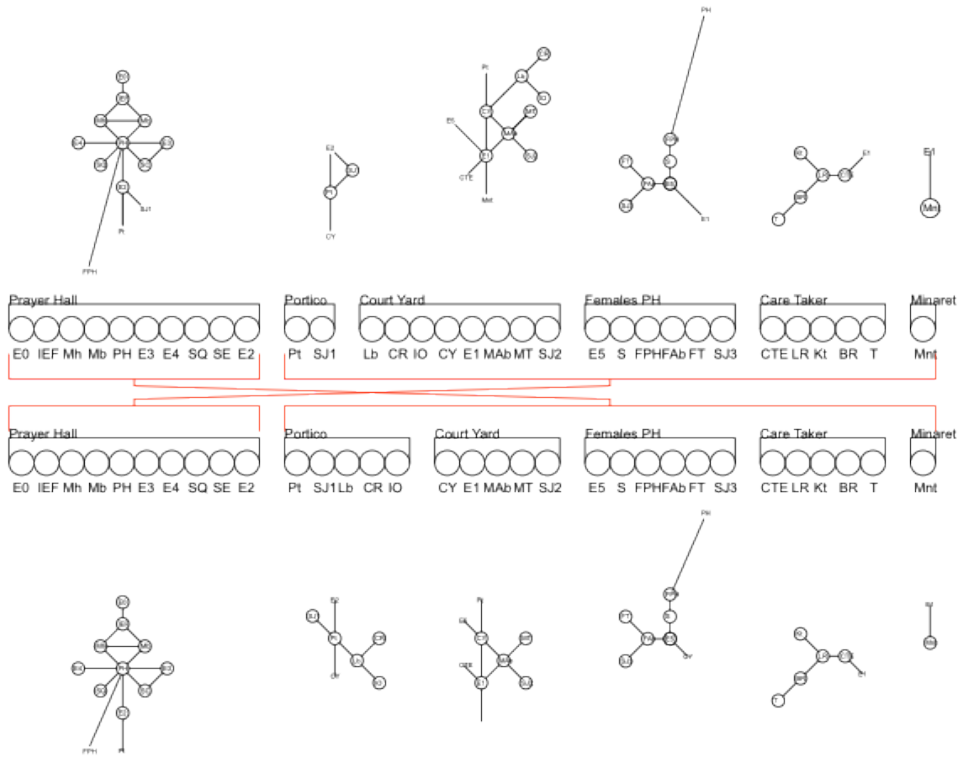


Figure 5-63 Genetic Operation

Genetic operation

Example showing a cross-over genetic operation between two parents on the Mosque zoning level.

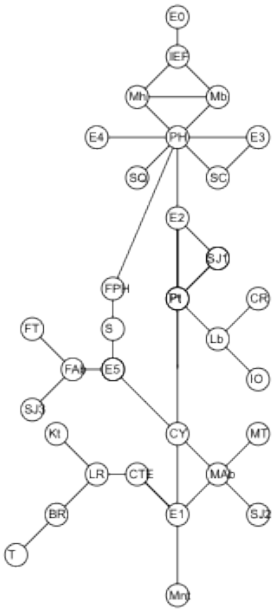


Figure 5-64 Sibling A

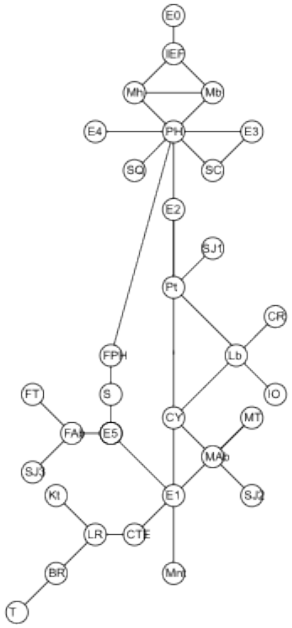


Figure 5-65 Sibling B

Evolved siblings

Evolved generation of different configurations. The genetic operation in this case, took place between the dual graphs of the two parents.

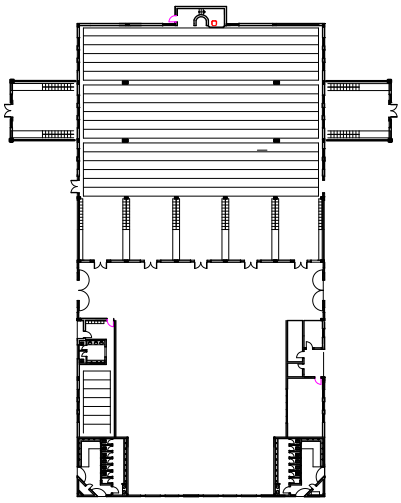


Figure 5-66 Layout variant 1

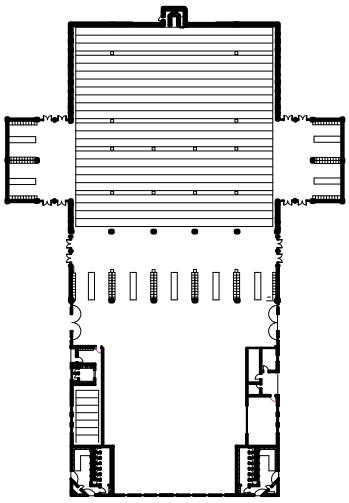


Figure 5-67 Layout variant 4

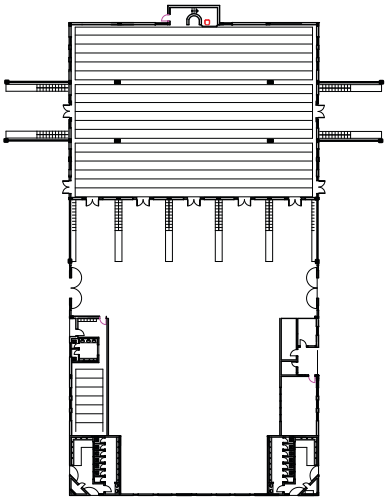


Figure 5-68 Layout variant 2

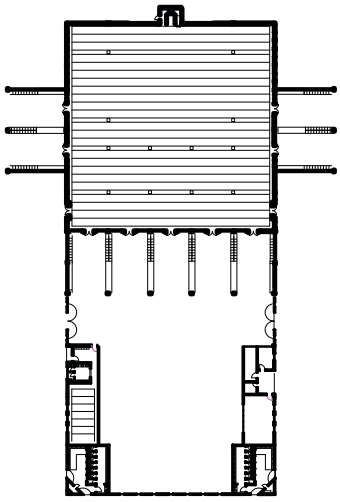


Figure 5-69 Layout variant 5

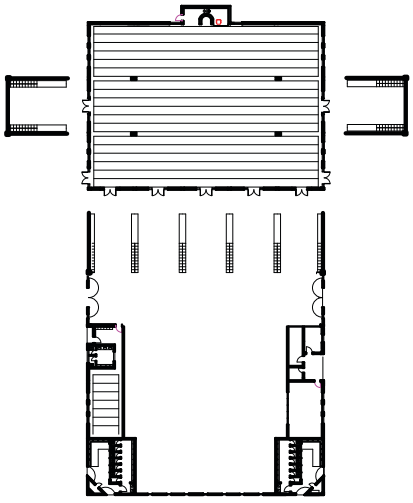


Figure 5-70 Layout variant 3

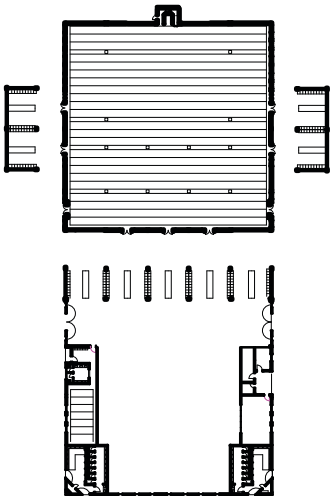


Figure 5-71 Layout variant 6

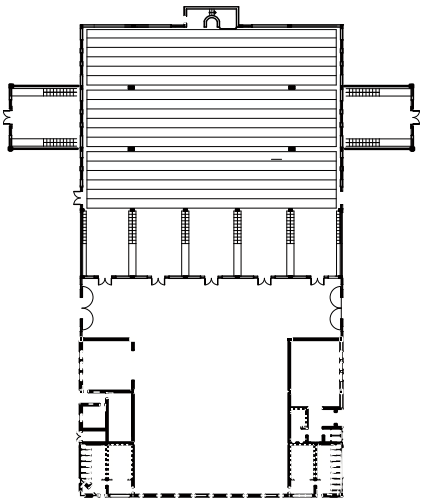


Figure 5-72 Layout variant 7

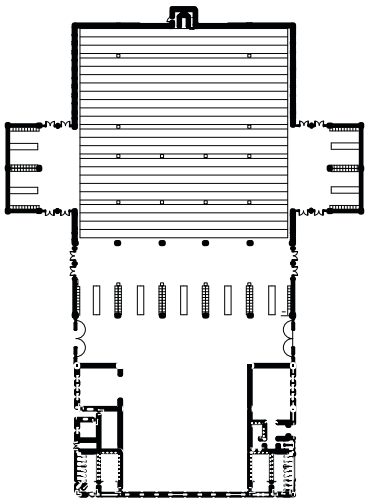


Figure 5-73 Layout variant 10

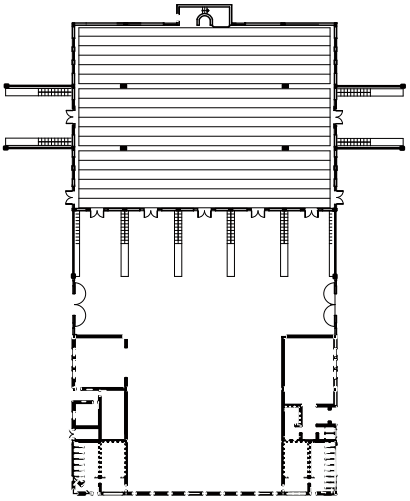


Figure 5-74 Layout variant 8

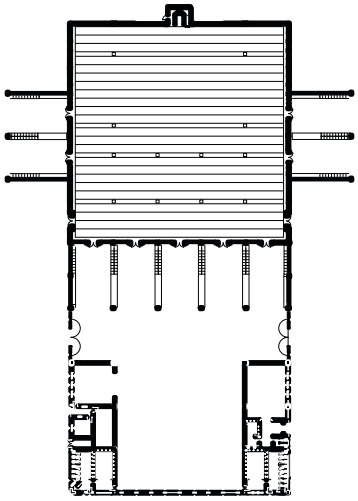


Figure 5-75 Layout variant 11

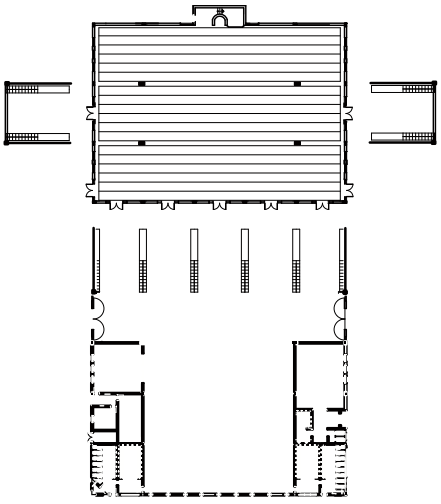


Figure 5-76 Layout variant 9

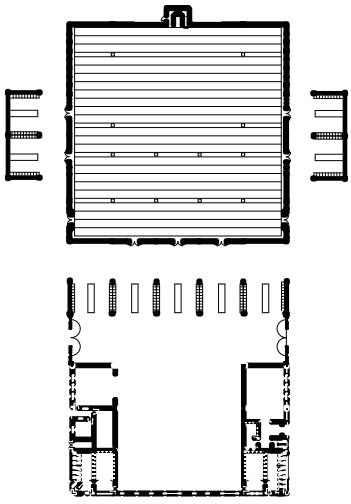


Figure 5-77 Layout variant 12

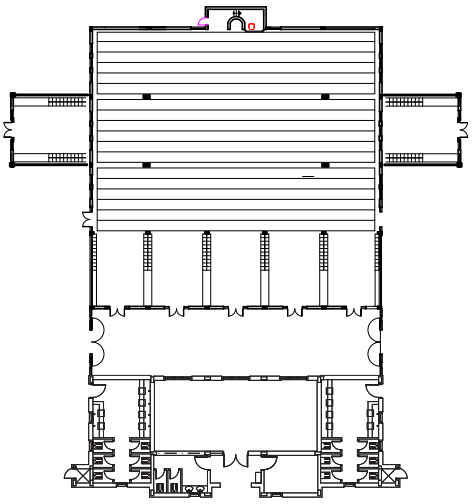


Figure 5-78 Layout variant 13

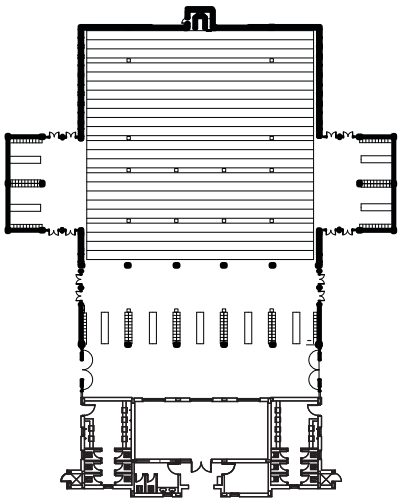


Figure 5-79 Layout variant 16

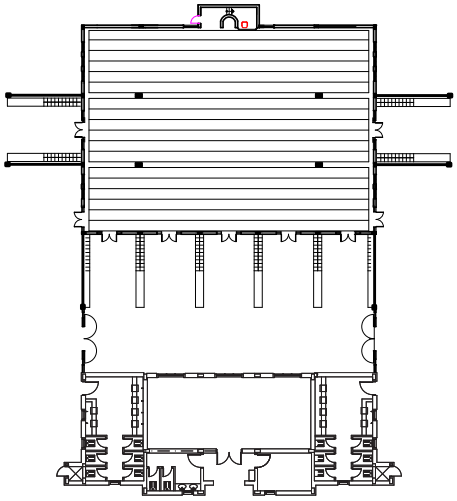


Figure 5-80 Layout variant 14

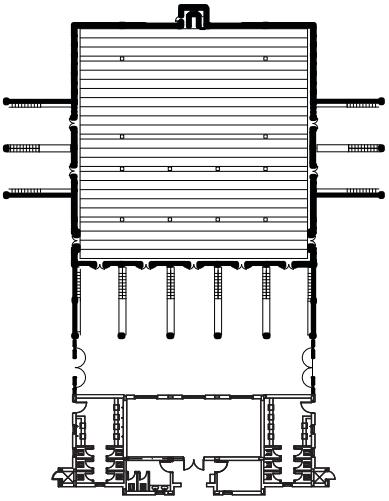


Figure 5-81 Layout variant 17

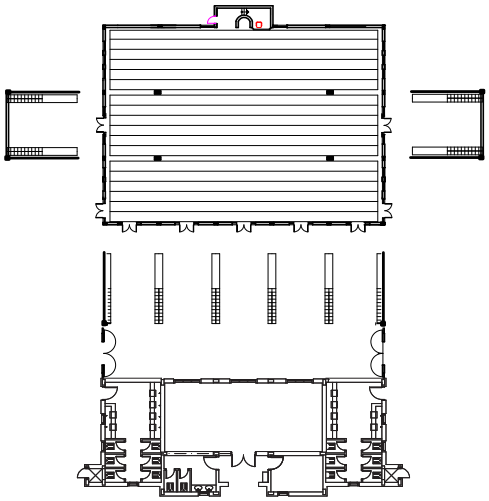


Figure 5-82 Layout variant 15

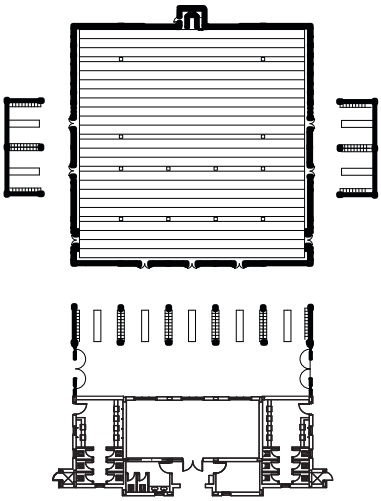


Figure 5-83 Layout variant 18

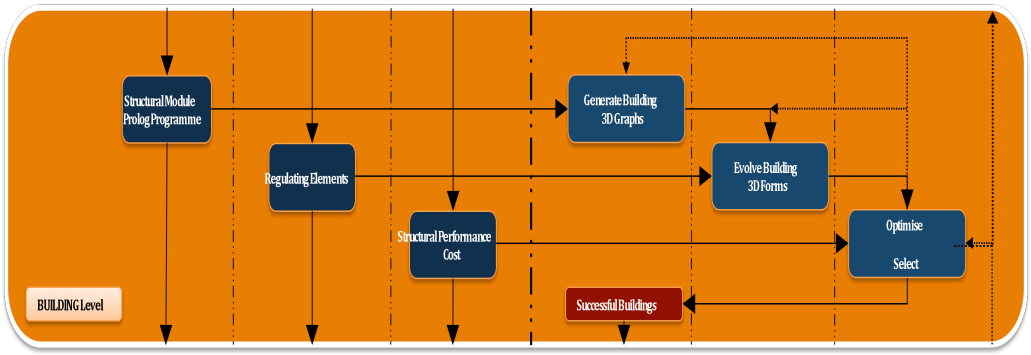
The diagrams in Figure 5-66 to Figure 5-83, illustrate the system generating a number of layout variants, using the three previously illustrated configurations

of the prayer hall (room) with other ancillary spaces (courtyard, ablutions area etc). The limited number of configurations used here is for ease in illustration only. The system will explore a vast number of possible layout configurations according to the number of the successfully generated room variants for each space.

5.2.2.2.6 Layout level analysis, evaluation and selection processes

Once a successful solution has been generated, all the variants for each room within that solution may not necessarily be the best solution. Thus, although the solution may be generated and considered a success, a higher overall score could potentially be achieved by selecting some different room variants. The HEAD system allows a retrospective 'fix' to successfully generated solutions by looping back to the previous levels. The evaluation of the how closely the generated layout responds to the regulating lines and adjacency requirements facilitate the decision on whether or not the generated solution has an aesthetic value and can be used as a supplementary 'fitness function' in the overall evaluation of the generated layout. The main fitness functions at this level, however, are based on functional performance and include such aspects as the total area of the building, building perimeter length and the cost.

5.2.2.3 Building level



5.2.2.3.1 Building level database

5.2.2.3.1.1 Structural form module

The designer can create a specially-designed modular structural component to interpret the chosen design concept. For the purposes of this illustrative paper-based simulation a study was conducted to abstract the

modular structural component of each of the historical Mosque typologies Figure 5-84 to Figure 5-89. The structural module can, however, be based on any criteria. For example, in the case of the mosque's features or its responses to particular structural design constraints could have served as the basis for the structural module design.

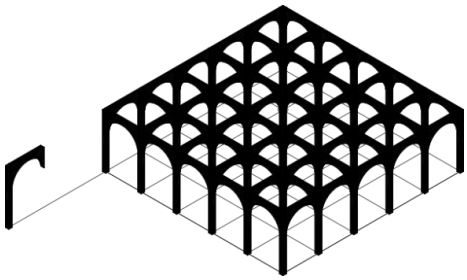


Figure 5-84 Hypostyle typology, structural component and formation

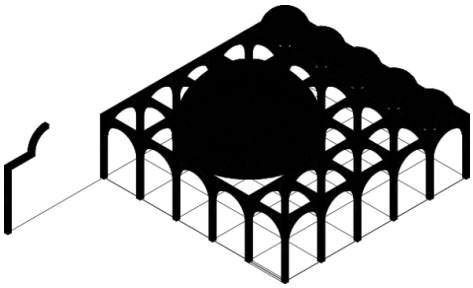


Figure 5-85 Hypostyle with Dome Accent & Central Dome typologies, structural component and formation

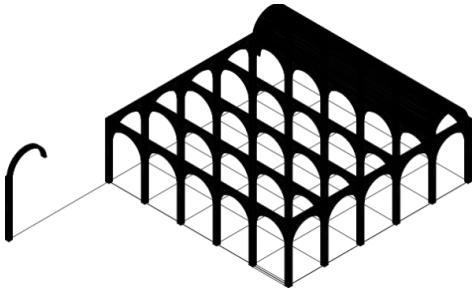


Figure 5-86 Hypostyle with Domical Vaulting typology, structural component and formation

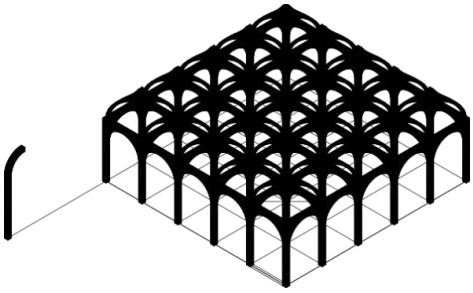


Figure 5-87 Hypostyle with domical Vaulting typology, structural component and formation

Figure 5-88 Iwan structural component

Figure 5-89 Crossing Vaults

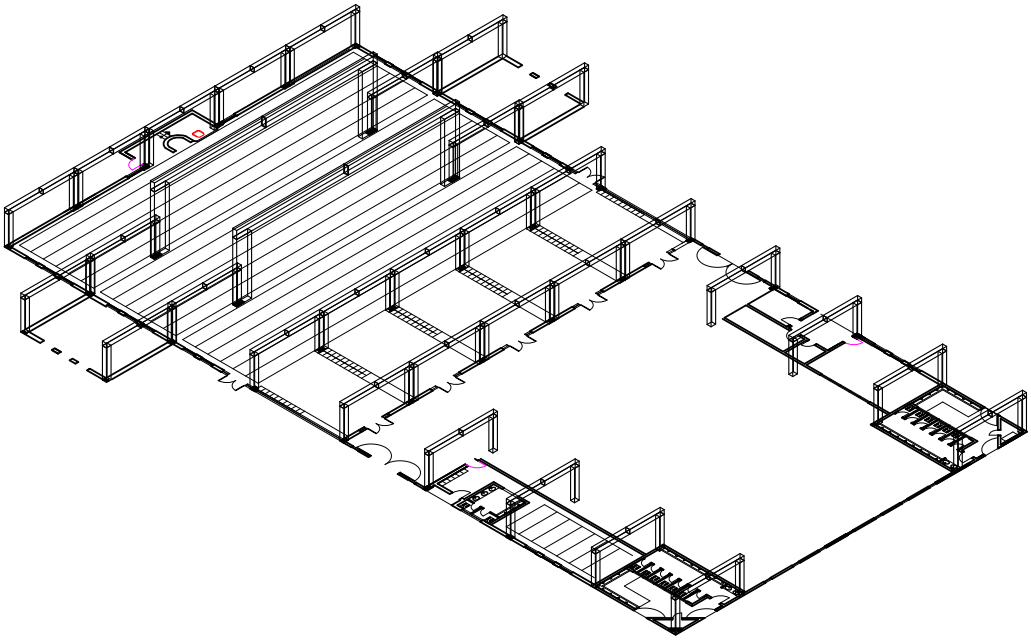


Figure 5-90 Building variant 1

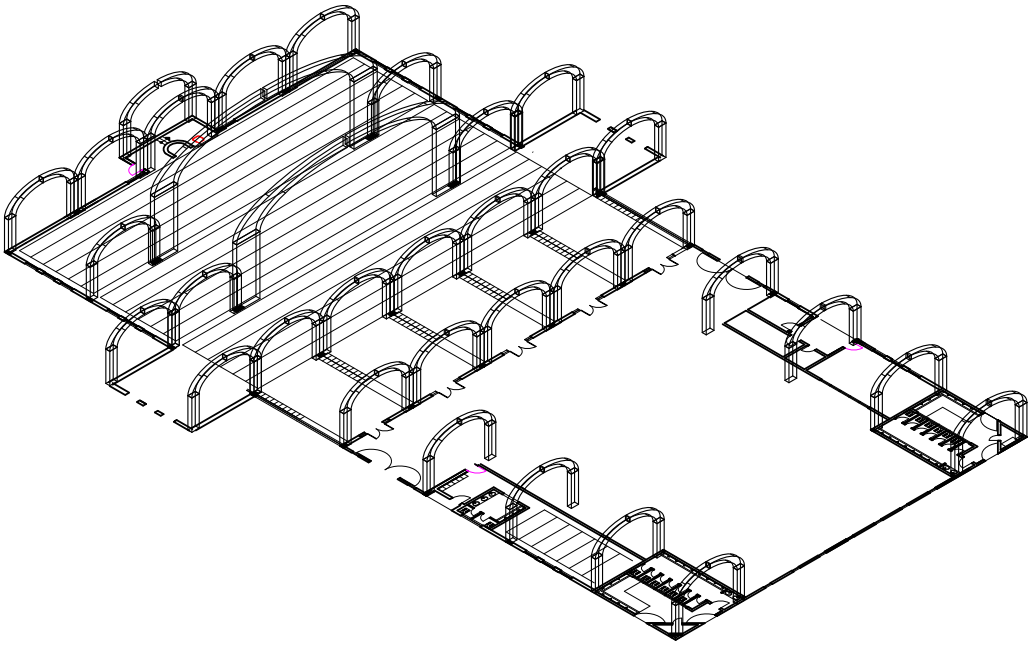


Figure 5-91 Building variant 2

To demonstrate the insertion of the structural components into the successfully generated layout, Figure 5-66 Layout variant 1, is used here with two different structural components; Figure 5-90 a simple column and beam structure is used similar to the structural component of the hypostyle hall typology. Figure 5-91 a column with an arch is used similar to the structural component of the hypostyle with domical vaulting typology.

5.2.2.3.1.2 Prolog prescription

High level descriptions of various mosques were developed using definite clause grammar (DCG) to build seeds from high level descriptions of various mosques, resulting in a symbolic representation of sequence strings out of the different configurations, to govern the building level algorithm, see section (7.4).

The spatial organisations were graphically represented according to the adjacency constraints between the spaces. The designer, using DCGs and parse-trees, was thus able to prescribe the spaces and relationships between the spaces within the building using Prolog. Once the parse-tree had been generated by the Prolog program, it was rendered in a dual graph as a graphical representation of the spatial organisation of the building's spaces. Each configuration was represented by a graph where nodes were placed to represent spaces that were linked by relationship lines analogous with graph and circuit theories.

The spatial configuration represented in the dual graph served as an automatable depiction that could be fed into the genetic algorithm system. This transfer into the genetic algorithm system could be achieved by translating every dual graph into orthogonal rectangular duals, which would then be encoded into the genetic algorithm system with an expected performance criteria set to govern the evolution. The adjacency matrices and reproduction operators of the graphs could be topologically configured and encoded for use by the genetic algorithm system to allow for further exploration of the larger space of graph topologies.

5.2.2.3.2 Building level consistency controller

5.2.2.3.2.1 Regulating elements

The massing of the overall building is based on the configuration of the structural components. The regulating elements, which include the regulating lines, points, planes and volumes generated out of the regulating lines and their points of intersections, govern the building's massing configurations. In addition, the transformational rules of the parameterised

structural module indicate the right insertion points for the structural components into each of the successful layouts. The generated layout of the spatial configuration, transformed by the parameterised modular structural component into a three-dimensional form is, in effect, governed by the regulating elements.

5.2.2.3.3 Building Level fitness functions

- Structural performance analysis

As discussed in section (4.6.4.2)

- Cost

As discussed in section (5.1.4.3.4.2)

5.2.2.3.4 Building level generative transformational algorithm

The transformational algorithm allocates insertions points and lines for the building's structural components and walls (or other three-dimensional objects). The algorithm uses the regulating elements as a reference point and is based on the Prolog prescriptions of the building's configuration, which are translated into a parse-tree and grammatical strings to govern the Synthesis of the building's structural formation. The translation of the Prolog prescriptions into the parse-trees is guided by the regulating elements. The regulating elements, in their role of guiding the massing configuration, can be seen as the basis for the expression of the building as a whole.

The structural components, which have transformational rules and relations assigned to them, can permutate into various alternative structural skeletons for the same layout. Every possible structural skeleton for every possible layout can be translated and depicted as a three-dimensional rectangular dual graph, which can be used as the initial (parent) generation for the evolutionary algorithm to operate on.

The transition from the transformational algorithm system to the genetic algorithm system occurs when every dual graph is translated into an orthogonal three-dimensional rectangular dual. Topological configuration and encoding of the adjacency matrices and reproduction operators of the graphs can be carried out so that they can be used by the genetic algorithm system, which will ultimately facilitate larger spaces of graph topologies to be searched.

5.2.2.3.5 Building level evolutionary genetic algorithm

The initial 'parent' generation of the building level evolutionary genetic algorithm consists of both identical layouts that differ in the structural configuration and the different layouts with the different structural configurations. Genetic operations can take place between any of the parents.

5.2.2.3.6 Building level analysis, evaluation and selection processes

As is the case at all levels of the HEAD system, solutions generated at the building level may need to be optimised after evaluation. If, after optimization, the problems highlighted by the evaluation cannot be solved at the building level, the system can loop back to the earlier level that is relevant to that particular problem and attempt to solve the problem. The solution is only discarded if attempts to solve the problem have been unsuccessful at all levels.

Aesthetic 'evaluation' is provided once again by the solution's compliance with the regulating lines and elements that emerge from the earlier levels of the system. The regulating elements also provide a guide for ordering and generating the solutions at the building level. The evaluation of the building level solutions as a whole, in other words, the propagated models, is based on the structural performance and behaviour and can be facilitated by the use of structural performance software.

5.2.2.4 Optimisation level

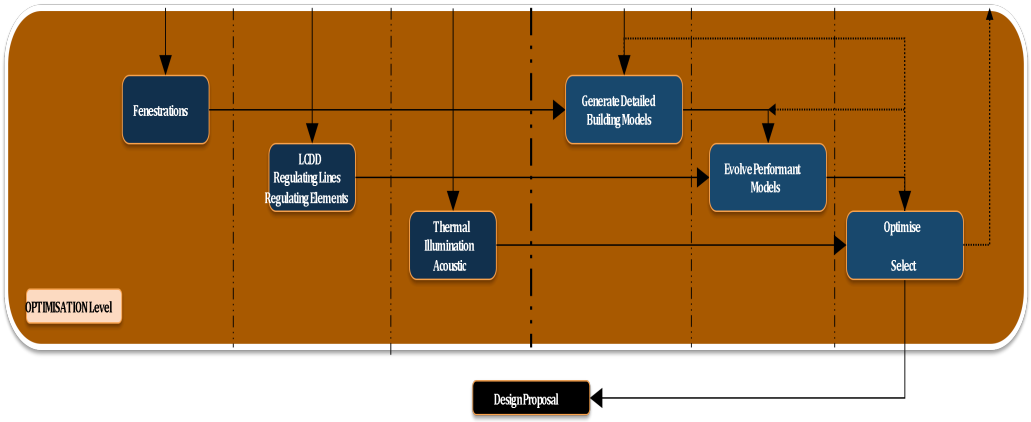


Figure 5-92 Mosque's optimisation level

5.2.2.4.1 Optimisation level database

The optimisation level database is comprised of the accumulated information from all levels of the system. In addition the desired and expected internal performance of the building can also be entered into the optimisation database. The environmental performance requirements of the building, for instance, the thermal, illumination and acoustic requirements of each space within the building and the building as a whole, are governed by the functional requirements of the building. The HEAD system can analyse and evaluate each generated model against the requirements and will highlight the areas that require further optimisation.

The optimisation level database can store mathematical formulas predicting human behaviors in various scenarios animating the occupants' sequential activities as the performance criteria against which the generated models can be evaluated. The building information that can be evaluated by the system against environmental performance, at this level, can take the form of data for the fenestrations, for example. The window types and shapes can be selected and then parameterised by setting the proportional relationships between the window variants.

5.2.2.4.2 Optimisation level consistency controller

The consistency controllers of each level of the system are maintained in the optimisation level. The combined set of consistency controllers guides the permutation and propagation processes that evolve solutions

conforming to the requirements of all previous levels. Any changes made to even the smallest aspect of the evolved solutions during the optimisation processes can affect the whole solution (including the aspects which were successful). The consistency controller sub-component of the optimisation level can highlight which other aspects (or objects) of the solution will be affected by the change and also, how it will be affected. The consistency controller sub-component thus enables the system to keep track of any changes made and facilitates the resolution of any problems ideally without causing further problems.

5.2.2.4.3 Optimisation level fitness functions

The fitness functions that are used to select the 'best' members of the population generated by the system during early design are the heat flow for the four solstice and equinox dates, the cost as a function of the sizes of the various building components, acoustic performance for the prayer hall, occupants' behaviours and CO2 equivalent emissions of the building materials. While the EnergyPlus simulation program can be used for the analysis and evaluation of a number of aspects of the building solution, other performance simulation programs would be used in conjunction. For instance, building regulations and codes, design standards and cost analysis, when linked to the evolutionary design system, will allow for a far wider range of fitness functions against which the generated solutions can be evaluated. Mathematical models predicting human behaviours can be used in the animation of he architectural design. The mathematical models could be inserted as a genetic algorithm fitness function.

Experiments were conducted to test the approach. These performed optimisation by evaluating the fitness against thermal, cost and environmental performance for each individual in the population. Weather information for specific locations, such as thermal performance during heating and cooling days, wind effects and environment factors were compiled from weather data files. The following are some fitness functions developed to calculate those different factors on addition to a weighting factor to calculate the overall performance according to different priorities:

- Thermal algorithms

  Heat Factor, function of heat flow through envelope

  $HF = \Sigma U_{ext} \times A_{ext} \times (k_{heat} \times DD_{heat} + k_{cool} \times (DD_{cool} - NTC)) \times OF$

  Base on heating (DD_{heat}) and cooling (DD_{cool}) degree-days

  Weight heating (k_{heat}) & cooling (k_{cool}) separately

  Night-time cooling (NTC)

  Orientation factor (OF) allows for direct & diffuse radiation

- Cost & Environmental

  Total element area * element impact

  Cost Factor, $CF = \Sigma A_{ext} \times Cost$

  Environment Factor, $EF = \Sigma a_{ext} \times CO_2\text{-e}$

- Weightings on each factor to allow for local variations

  $Fitness = W_h \times HF + W_c \times CF + W_e \times EF$

5.2.2.4.4 Optimisation level generative transformational algorithm

The fitness functions that are used to select the 'best' members of the population generated by the system are the heat flow for the four solstice and equinox dates, the cost as a function of the sizes of the various building components, acoustic performance for the prayer hall, occupants' behaviours and CO2 equivalent emissions of the building materials. While the EnergyPlus simulation program can be used for analysis and evaluation of a number of aspects of the building solution, other performance simulation programs can and should be used in conjunction. For instance building regulations and codes, design standards and cost analysis, when linked to the evolutionary design system will allow for a far wider range of fitness functions against which generated solutions can be evaluated.

Mathematical models predicting human behaviours can be used in the simulation of the architectural design. The mathematical models could be inserted as a genetic algorithm fitness function.

5.2.2.4.5 Optimisation level evolutionary genetic algorithm

The operations of the evolutionary genetic algorithm at the optimisation level can take place between parents, which are, in effect, the same model with different fenestrations or between parents that are different models and thus have different fenestrations. In the case of parents that are of the same model (but with different fenestrations), the propagation will result in the same sibling model, but each with different facades. If, on the other hand, the parents are different models (again with different fenestrations), the siblings of the different models and consequently the different facades will be propagated.

5.2.2.4.6 Optimisation level analysis, evaluation and selection processes

The Analysis, Evaluation and Selection sub-component of the optimisation level of the system highlights, guides and directs the inner optimisation of all successfully-generated solutions that require minor optimisation. The HEAD system is ordered hierarchically, thus all the data used in the generation of the successful models is accumulated throughout the design process. The analysis, evaluation and selection sub-component, using this accumulated data and referring back to the generative transformational and evolutionary genetic algorithms of the previous levels can thus attempt to optimise any models requiring optimisation. Only if this optimisation is unsuccessful will the system loop back to a previous model where the major changes can be implemented.

In the event that a model does require major optimisation during either the permutation or the propagation of the design solutions, the system can resolve the problem by looping back to the earlier level where the problem originated. Another option available is for the system to search the priority settings and make changes accordingly. In cases where a problem cannot

be solved at any level and the conflicts remain, the solution can be deleted or the designer can make a subjective decision on the particular solution in question.

6 Discussions and Conclusions

As stated earlier, the overall research project set out to develop a design method and system for architectural building design in which the designer's human cognitive process and creativity take a central role within the Computer Aided Architectural design (CAAD) process. The research proposed that the transition between the designer's cognition and the computer's closed-logic systems could be bridged with a design schema. The hypothetical proposition around which the research was conducted was based on the notion that an integrated interdisciplinary design approach could be achieved by combining the generative evolutionary design system with performance simulation systems.

In this chapter the findings of the 'Conceptualisation' and the 'Developmental' stages which, are the scope of work of this thesis are discussed and the conclusions that are drawn from them are presented. We begin with a statement of the results and put these into the context of the research by highlighting the theories underpinning the main areas of the research. The research methodology that was adopted is summarised so that the results can be understood in context. Particular aspects of the research are highlighted that are considered to be significant. This is followed by a discussion of the importance of the research as a whole. The next section identifies the challenges that this research project faced and also highlights the potential challenges for the further development of the proposed design method and the HEAD system. A conclusion follows this section and finally the last section highlights areas for further research.

6.1 Statement of results

The designer's creativity and preconceptions can be interpreted and encoded into computer logic using an algorithmic design approach and the design schema applies the notion of the tectonic process-oriented design method.

The parametric design approach takes advantage of the benefits of standardisation in building design without violating the architectural principles of the uniqueness of each project.

The use of modularity within parametric design can be seen as a transition between the conventional design method and the digital design method taking advantage of both.

Generative design systems using space layout planning methods have the potential for supporting the genetic evolutionary systems by producing an initial generation for the evolutionary system to begin the optimising and thereby begin the process of evolving successful solutions.

Evolutionary architecture has the potential of resolving multiple conflicting design constraints using fitness functions. The assessment of the generated solutions are based on these fitness functions and and the trade-offs between them. Thus, the true sense of a multi-disciplinary integrated design method is achieved.

The Hierarchical setup of the HEAD system reflects the decomposition of the design problem and the gradual synthesis of the design solution from one successful design state to the next according to a prioritized requirements, highlights the huge potential for the use of the evolutionary design system to adapt different design approaches successfully.

6.1.1 Summary of the theory underpinning the research

The theoretical background related to this area of study has been discussed in depth in previous chapters. In this section, the main theories underpinning this research and the development of the proposed design method and system architecture are highlighted in a brief discussion of the basis upon which the research progressed. Creativity in Design, Algorithmic Design Approach, Modularity Design Approach and Integrated Design Approach are the main areas under which the relevant theories were categorised.

6.1.1.1 Creativity in design thinking

In developing a design method and system, regardless of whether it is computer aided or not, a vital starting point is in gaining an understanding of how human designers think and how design evolves. Although a great deal of research conducted over the years has provided us with a general understanding of human creativity and human cognition in design, there is no single accepted definition for either.

As discussed earlier, human creativity in design is not clearly understood and remains mysterious. Yet existing CAD applications attempt to simulate the human design activity. The result is that the current systems are found more useful in the production of the design rather than in supporting the actual design process.

Existing CAD software depend on precise geometric information in order to represent a design model. Early phases of the design process are characterised by an exploration and modification of alternatives that means that the solutions and sub-solutions produced are indefinite.

Evolutionary techniques can be used to explore the solution space of an ill-defined problem, as in the case of design problems. At the same time, genetic algorithms support the analytical evaluation of generated solution candidates, which means that the successful solutions are justified through objective quantifiable measures.

6.1.1.2 Algorithmic design approach

This research acknowledges that certain aspects of the conventional design method cannot be discarded even with development of CAD. The school of thought that advances the notion that CAAD should attempt to imitate (or, at the very least, follow) conventional design methods is rejected, as is the opposing school of though which argues that a new design process should be developed, based purely in the domain of computer science. A more balanced position that recognises the need to incorporate elements of both conventional and computer-aided design methods is taken.

The proposed design method and the HEAD system are largely based on the use of algorithmic design approaches to bridge the gap between the human mind and the computer, because the inductive logic of an algorithm can be seen as a computer's creativity (Aranda & Lasch, 2006; Terzidis, 2006). The position adopted by this research is that digital design need not be constrained by attempts to follow every aspect of conventional design approaches, neither is it seen as necessary to completely abandon the conventional approaches. Instead, the research and subsequent development of the proposed design method and the HEAD system are based on the notion that digital design can benefit from incorporating certain select elements of conventional design, such as the designer's creativity and preconceptions, in the domain of the computer.

6.1.1.3 Modular design approach

Through a parametric design method, this research incorporates a modular approach to design. Parametric design uses a variable set of parametric rules and mechanisms, giving the designer a range of diverse and unique building components that can be used to create a module (Agkathidis, 2009). The fact that the designer can define the parameters (which include variable geometrical definitions and behaviours) of the design means that he controls the model that results from the modules. Not only does the parametric design method enable the designer to define the relationship between the various design components of a parametrically-defined model, but it also allows him to set the constraints or rules that govern the defined variables. As Shah and Mäntyla (1995) point out, this means that the produced model can be manipulated while control of the design itself is maintained throughout the design process.

This research incorporates the idea that through a parametric design method, a modular approach to design, without the limitations of exclusively having pre-designed components can be used. This research accepts the notion that because both the components and the assembling techniques used in this kind of modular design can be created and set by the designer, a certain level of standardisation in the design process can be

achieved while maintaining the artistic freedom of the designer. While various components of the building can be standardised, the building, as a whole need, not be as the constituent (standardised) parts can be assembled using the evolutionary design approach, in response to the environment.

6.1.1.4 Integrated design approach

In a general sense, architectural design seeks to find the most balanced solution to the physical, visual and performance requirements of a building. As Bachman (2003) points out, building designers attempt to integrate the aesthetic and artistic elements of the design with the scientific realities of their environment. Traditional, conventional design methods ordinarily followed the process of initially producing a design solution that would subsequently be tested against performance criteria. More and more, this retrospective performance assessment is seen as not only a waste of resources, but is also seen as limiting the responsiveness of the design solution to its environment, because it depends on the designer's subjectivity as drawn from previous experience. In response to these limitations, integrated design approaches are being widely advanced.

This research acknowledges the importance of integrated design approaches with the understanding that the requirements of the multiple disciplines should be included into the design process at the earliest stages, thereby enabling the designer to produce functionally responsive, sustainable and economically sound design solutions. Rather than assessing the produced designs at the latest stages and be forced to re-visit the earliest stages of the design process, an integrated approach can help highlight potential areas of weakness that require improvement early on, thus reducing the cost-and-time requirements. This can be achieved by the use of the assessment capabilities within the evolutionary architectural genetic algorithm. Different fitness functions can be assigned and while each design solution aims to achieve the highest score against each fitness function, the solution with the highest combined score covering all fitness

functions will represent the most successful multi-performant, interdisciplinary solution.

6.1.2 Review of the methodology

The research methodology adopted by this research project is discussed in detail in Section (2.4). What follows in this section is a brief review of the most salient points.

Because the subject matter of this research spans a wide range and variety of sub-subjects, a linear research approach was felt to be too limiting. In dealing with a research project of this size and scope 'methodological pluralism', which incorporates multiple theoretical models in addition to the multiple methodological approaches in scientific practice (Norgaard, 1989; Polkinghorne, 1983) was seen as the most beneficial research method. This combination of the research methods acknowledges the importance of both prescriptive and descriptive research and uses each during different stages of the research project where they are most suitable.

The research method adopted here is composed of the three key stages of research that result from the combination of Blessing's (2009) *Design Research Methodology* and Nunamaker et al.'s (1990; 1990) research process. The three research stages identified are 'conceptualisation', 'development' and 'evaluation'.

The literature review of the relevant materials was conducted during the 'conceptualisation' stage and clarified the main areas within the wider overall subject that would be researched in more detail. The potential areas in which the research could make a contribution were also clarified at this stage. Included within this 'conceptualisation' stage, once a clearer picture of the research area was obtained, were set the firm goals and objectives that the research intended to follow. As a final part of the 'conceptualisation' stage, an exploratory review that facilitated the development of a conceptual design method was carried out.

During the second stage of the research method that followed, the 'developmental' stage, the conceptual design method was further developed into the proposed HEAD system. This was achieved using a prescriptive approach. By specifying the system components, (with their structural relationships and dynamic interactions) and the functionality of the system at this 'developmental' stage, the future building of the proposed design system would be facilitated during the next stage of 'Evaluation'. The developmental stage also included the setting up of an illustrative paper-based simulation, the results of which are expected to aid in the eventual assessment of the HEAD system's data-structure, databases and knowledge bases. A standardised building type was selected for the illustrative paper-based simulation with the view of simplifying the eventual and ultimate assessment of the proposed design method — that of its feasibility.

The last research stage identified, the 'evaluation' stage, is beyond the scope of this project. This stage would be composed of an assessment and evaluation of the research project as a whole. Ideally, the evaluation of the proposed design method would be carried out by building a prototype system that would prove the proposed design system to be both feasible and usable.

6.1.3 Illustration of research findings

Because the building and evaluation of the proposed design method and system is not a part of this research project, the results can be seen in terms of the outputs from its conceptualisation and developmental stages. The main outputs have been identified and the importance of each is discussed below.

6.1.3.1 Research question and hypothetical proposition

How can generative and evolutionary design promote an integrated, interdisciplinary building design approach to support multi-evolutional criteria while fostering a collaborative relationship between human creativity and computer capabilities at multiple levels of detail?

The research question and the hypothetical research proposition, which reflects the research goals and objectives, are seen to offer a valuable insight, because they open up new avenues for further research into CAAD. The research proposition highlights CAAD systems that have undergone great developments, but have not yet reached their full potential. By identifying the advances in the various fields of building design research, which have the greatest potential for aiding in the production of sustainable performant designs and proposing a way in which to adapt and integrate these with each other, the research question offers a way to produce a single optimised design method and a system that incorporates the advantages offered by these advances.

6.1.3.2 Design method

The conceptualisation of the design method and the HEAD system, in a way in which the schema can be encoded and then used by the generative and evolutionary systems, is supported by the theories underpinning this research. This is, of course, a positive outcome for the research project, because it indicates that the outcome is viable. The design method conceptualised by this research project is also seen as a very important outcome that forms the theoretical foundations for the development of the proposed system.

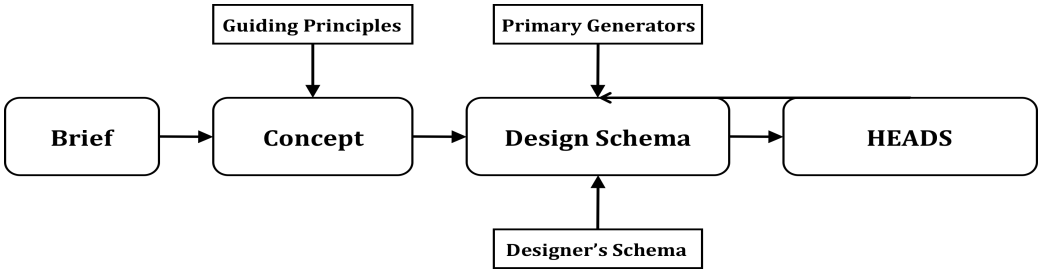


Figure 6-1 Conceptual design method

6.1.3.3 Design schema

The proposed design method adopts the design schema as the intermediate interpreter of the designer's creativity. The design schema, as envisioned by this research, is a formal expression of the design goals and constraints that allows the designer to generate rules that then generate the design. The design schema is shown to successfully interpret the open logic system of the designer's creative vision into the closed logic system of the computer. The use of current CAAD systems represents a trade-off between the advantages gained by computerisation and the human creative impulse. In the case of the proposed design system however, computation rather than computerisation is used. This means that the designer is aware of the embedded processes that the computer system will run and is thus able to interpret and translate his creativity directly into the computer without having to use a transitional medium (such as a sketch or a conventionally-produced partial solution), as is the case with computerisation. The proposed design method is thus seen as offering an important step forward in the use of the great benefits of computation without compromising the designer's creative input into the design process.

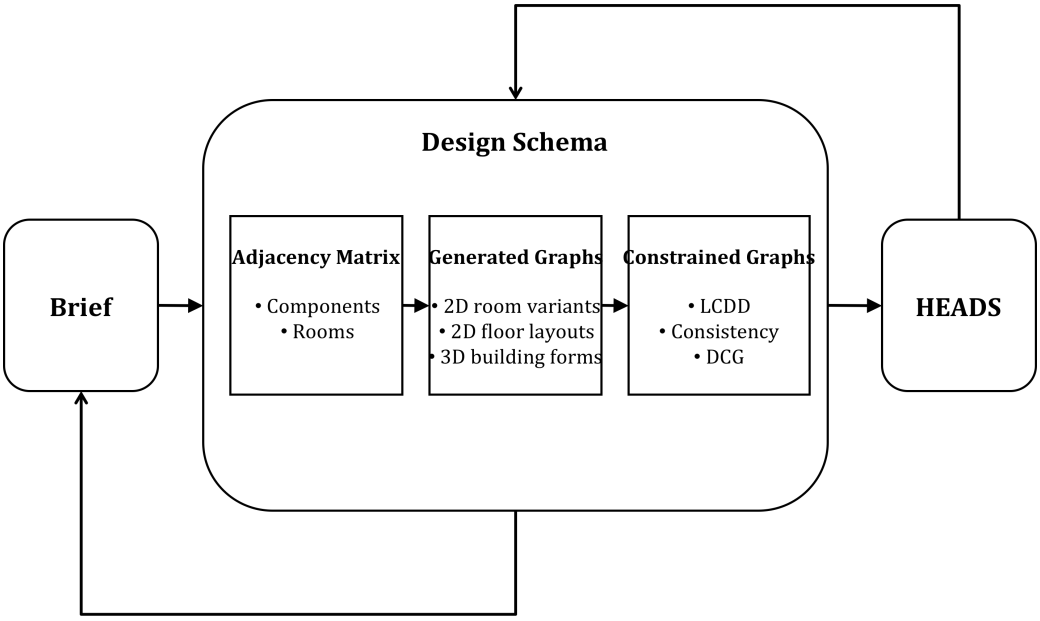


Figure 6-2 Design schema

6.1.3.4 The Synthesis Algorithms

The Synthesis Algorithms component, with its multi-level hierarchical approach to generative and evolutionary design, assigns multiple fitness functions for the various disciplines within the design process at each of its levels. The hierarchical set-up of the system represents a modular approach to design and allows the design problem to be decomposed so that at each level of the system, only targeted outcomes are produced. This is seen as an important outcome, because the design solutions produced at each level are responsive to the requirements of that specific level. The multiple-fitness functions (which are prioritised in order of preference) represent an integrated, interdisciplinary approach and are used to evaluate the solutions produced at each level of the hierarchical design system. As with the hierarchical nature of the system, these interdisciplinary fitness functions used at each level of the system are considered to be an important development, because non-viable solutions will be discarded at each level, thus reducing the number of outcomes that will need to be processed by the next level.

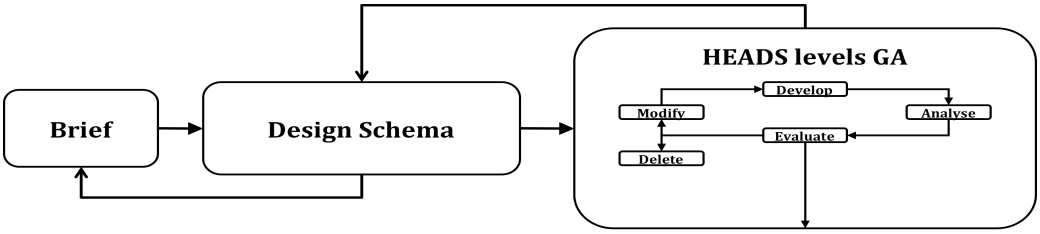


Figure 6-3 HEADS

6.1.3.5 The HEAD system

The main output of this research project is the development of the computational architecture of the HEAD system. The proposed system is the culmination of the conceptualisation stage of the research project. It is, in effect, a translation of the initial conceptual ideas of the research and represents the achievement of the research project's goals as outlined in the research proposition.

From a wider perspective, the detailed HEAD system architecture, developed to reflect the conceptual design method, can be seen as the blueprints for a viable computer design system. This HEAD system architecture is seen as an important result, because it describes the functionality and requirements of the system components in addition to the interrelationship between these components. The computational architecture of the system illustrates the system's design processes and procedures.

6.1.3.6 Illustrative paper-based simulation

While the illustrative paper-based simulation represents an initial conceptual implementation of the proposed design method and system, it simultaneously served as a facilitator in the developmental stage of the research. The research project did not take a simple linear path and, as mentioned earlier, an action research methodology was used. This meant that, while the illustrative paper-based simulation was actually in the process of being designed, it was possible to re-visit both the proposed design method and the HEAD system, and incorporate the findings from the illustrative paper-based simulation back into the development of the

system architecture. A very important and beneficial aspect of running the illustrative paper-based simulation in parallel with the other elements of the developmental stage is that the 'testing' actually contributed to the development of a more robust and viable proposed design method and system. This is because any weaknesses and issues identified by the illustrative paper-based simulation (the 'test') were immediately rectified in the proposed design method and system.

Ultimately, the results of the illustrative paper-based simulation, which was a manual implementation of the HEAD system, suggested that the proposed system is viable in terms of the achievement of the goals and objectives of the overall research project. The results also suggested that the HEAD system has the potential of being built and used with success.

6.1.4 Comparison with previous research

The HEAD System architecture is conceptualised on the existing research on both generative and evolutionary design techniques (Frazer, 2002) (Chan, 2008; Janssen, 2004; Liu & Frazer, 2002; Sun, 2002; Tang & Frazer, 2001). The HEAD System develops key aspects and introduces some new ways of working from these techniques.

Sun's (2002) system was developed to support product design and was tested and implemented on a mobile phone design around the concept of 'formatives' as outlined in section (3.4.3). The binary structured code script of Sun's system allows for a degree of flexibility and reusability of the system in designing different products. In the case of building design, the number of building components and the interdisciplinary relations between them, make the design process more complex than designing a product such as a mobile phone. Encoding the formative configuration for the genetic algorithm in a domain-specific representation might not efficiently generate unpredictable, novel and challenging designs, as discussed in section (4.7).

In comparison to Sun's previous work in this area of design, the HEAD System adopts a new task-specific representation for the domain of building design, where each one of the first three levels of the system deals with a different method of representation, leading ultimately to the formation of the overall solution. The HEAD system incorporates the generation of alternatives in the first three levels. The generated solutions will be assessed based on the assigned fitness functions according to the set targets in each level. The fourth optimisation level conducts an overall optimisation of the successful candidates accumulatively generated from the previous levels.

The concept of primitives, rudiments and formatives of Sun's system could be an effective decomposition of a configuration. To compare this with the hierarchical structure of the HEAD System, we can consider room components as rudiments composed out of primitives of basic geometries. A room, in that sense, would be a formative constituted out of rudiments of room components. Following the same structure, a layout is a formative composed out of rudiments of rooms that are composed of primitives of components. Furthermore, a building is a formative that is composed of rudiments of layouts that composed of primitives of rooms. This reflects the accumulative notion of an overall solution from sub-solutions of the decomposed design problems.

Table 6-1 Comparison between Sun's 'Formatives' and HEADS components

Sun's decomposition	Primitives	Rudiments	Formatives
HEADS 1st. level	Basic geometries	Components	Rooms
HEADS 2nd. Level	Components	Room	Layout
HEADS 3rd. level	Room	Layout	Building

Starting from the components being parametrically constructed out of their basic geometries and building-up to rooms, layout and building the HEAD System algorithms always have references of rudiments and primitive of each configuration, i. e. formative.

Sun's system adopts an aggregation approach where weighted values of separate fitness function performance are summed up to an overall fitness value for each generated prototype. The HEAD System develops this further and adopts a prioritisation order to guide the selection algorithm where a conflict might occur. Additionally, the hierarchical setup of HEAD System makes the system capable of incorporating more fitness functions according to the goal, which each level attempts to achieve.

The generative and evolutionary systems used by Sun were also utilized by Janssen (2004). Janssen's system, like the HEAD System, sees the design process as two main related tasks – the defining of the design schema and the development of the design from this schema. Although the design schema in the HEAD System shares some features of Janssen's design schema, it goes beyond Janssen's notion that the design schema consists of the common features of a family of designs. The design schema of the HEAD System is developed to incorporate the synthesis of the sub-solutions of the decomposed design problem. Additionally the HEAD System addresses the complexity and the interdisciplinary nature of building design more comprehensively through the capacity to incorporate multiple fitness functions.

Chan (2008) also developed a design system using generative and evolutionary techniques. Chan suggests a hierarchical representation of elements and mechanisms to deal with the complexity of a design problem. Chan's system represents the whole design process through a network of evolving elements of sub-solutions that interact with each other in accordance with pre-set evolving mechanisms.

The hierarchical structure of Chan's system is similar to the hierarchical nature of the HEAD System, in that both systems tackle the design problem

by decomposing it into sub-problems. The solutions to each sub-problem can then be fed into the next level of the hierarchy so that the ultimate design solution is achieved.

In the case of the HEAD System however, the hierarchical set-up is reflected in both the representation and the structure of the algorithms through the utilization of multiple representations and multiple algorithms, each dealing with a specific task aimed at achieving a specific goal according to the level at which it is operating. This allows the genetic algorithm of each level to incorporate the fitness functions that are relevant to the specific task being tackled.

As discussed in the literature review section of this thesis, space layout planning using graph theory representation has been under research and development for a long time (Grason, 1968; 1970, 1971; Levin, 1964; Rittel, 1968; Teague Jr, 1968). Recently some research was conducted to explore the incorporation of graph representation within evolutionary algorithmic systems. Homayouni (2007) and Wong (2009) demonstrated the feasibility of using graph theory to generate successful building layouts based on adjacency requirements and distance.

The HEAD System also utilises graph representation but does so to a greater degree than previous research. In the case of the HEAD System graph representation is used to generate not only the successful layouts for the building as a whole but also for the generation of individual rooms. While Homayouni and Wong propose the use of adjacency requirements to constrain geometry, HEAD System develops this and at both at the room and the layout level the graph representations are constrained not only by adjacency requirements but also by various design techniques and design criteria such as regulating lines, LCDDs, design standards, building codes and regulations. A further development made by the HEAD System is in the introduction of the use of the 3D graph, which Teague (1968) presented, to generate building forms from successfully generated layouts by using parameterised structural components constrained by regulating elements.

Throughout the generation process of the HEAD System multiple fitness functions are assigned to regulate the generation of solutions targeted by each level, ultimately generating successful building models.

6.1.5 Importance and contribution of the research

One of the main contributions made by this research to the area of CAAD is that the gap between the designer and the actual computational powers of a computer is narrowed. The design method and system proposed by this research offers the possibility for a designer with little or no prior knowledge of algorithmic programming to benefit more from the powerful generative and evolutionary systems that are based on genetic algorithms. The use of computers as 'computerisation' tools in current CAD systems does not take advantage of the potential that the genetic algorithms offer.

The design schema presented by this research, integrated within the evolutionary genetic algorithm, offers a means of using the computational capabilities of a computer in the actual design process. The schema presented by Frazer (2002) and Janssen (2002) is interpreted by this author as a 'designer's schema', because it is a 'personalised design method or framework' built up throughout the designer's working career. The design schema presented by this research, on the other hand, is extended to convey the notion that the design schema is not only a personalised design method, but is also specific to each problem at hand. The design schema as presented in this research provides a means of digitally formulating the designer's interpretations of the design problem with his creative vision. The transfer of the designer's creative vision into a computer system has always been problematic and this research proposes a solution that could have a positive impact on CAAD. It is proposed that the designer's creativity can be translated into a modular design approach of parameterised building blocks, with Prolog descriptions of the building's configurations and regulating elements being used to govern and guide the production of the design solutions throughout the permutation and propagation processes.

By setting up the HEAD system in a hierarchical geometrical multi-level manner, this research highlights a way in which the production of unviable solutions can be avoided. This is seen as a contribution to the field, because the computational processes are minimised and the required time for processing solutions is thus shorter. The hierarchical approach presented by this research also allows multiple and sequential assessments, using fitness functions, of the decomposed sub-problems all the way through to the overall composed design solution. The holistic approach to the design process is regulated with consistency controllers, which are the base and reference for the design decisions made throughout the evolution process.

The major contributions made by this research to the field can be seen in the form of the HEAD system with its two components, the Design Schema and the Synthesis Algorithms. Each of the HEAD system components is characterised by a new approach in applying previous and/or existing knowledge and research findings. The hierarchical iterative structural representation allows the capture of constraint-generated solutions at an appropriate level for various algorithms, while the use of the graph aids in constraining the geometry. The 2D/3D graphs introduced into the genetic system provide additional constraints on the generated solution through the genetic algorithm, thus providing a more efficient mechanism for the genetic algorithm to handle geometry. The most significant contributions made by this research can be highlighted and summarised as follows:

Design Schema

1. Translates the planar modularity, building block relationships, adjacency requirements and structural modular configurations into a two-dimensional and three-dimensional rectangular dual graph to generate seeds with defined fitness functions.

2. Regulating elements are used to guide the generation of aesthetically accepted building layout and building models.

3. The Prolog program using definite clause grammar (pseudo code is developed) to generate seeds for higher-level descriptions of the building.

Synthesis Algorithms

1. A multi-level hierarchical approach of a generative and an evolutionary design system to avoid totally unviable design solutions to minimise computations and time.

2. A multiple fitness function of interdisciplinary integrating design approach with weighting and prioritisation of the preferences to resolve conflicting design constraints and performance.

3. Consistency controllers are introduced to regulate the design process and to generate specific design solutions.

Although the benefits of the overall research discussed above relate to the *potential* of the proposed design method and system, this research project produced some more immediate benefits that will ultimately greatly facilitate and further the overall research. Because this research project's scope of work covers the first two phases of the overall research, namely the 'conceptualisation' and the 'developmental' stages, the results and findings are critical for the overall research project in general and for the last stage of 'evaluation' in particular. While the evaluation of the proposed design method and system cannot be undertaken by this research project, the main outputs of the research do, nevertheless, represent some important advances that will contribute to the further development of CAAD systems. The results of this research project can be used as the basis upon which further development and the ultimate building of the actual system can be achieved. In effect, this research project serves as a proof of feasibility for the proposed design system and also acts as a launch pad from which it can be built. The hypothesis and the proposition are supported by the resulting conceptual design method and the proposed design system. The proof of feasibility is demonstrated through the

illustrative paper-based simulation of the HEAD system in the chosen project. In addition, the key aspects of the proposed design system were implemented to test their validity. These key aspects tested include the regulating lines, graph theory, Prolog programming and the use of multiple-fitness functions in the genetic algorithm (examples of which can be found in the Appendix).

6.2 Challenges identified

From the very inception of the research proposition, there was never an expectation that the designer and the software engineers would have an instinctive meeting of the minds. Owing to the inherent differences between the two disciplines, the very thought processes of the two are completely different. At the most basic level, one can consider this as architectural designers lacking a formal education and understanding of computer programming and programmers lacking a formal education and understanding of design. In a practical sense, this creates more complications than one might, at first, expect. A simple example of the challenges encountered because of this difference in the thought processes can be seen in the perception of a simple cube — while an architect will visualise the actual form or shape of the cube, a software engineers will process the cube as a series of coordinate points and connecting vectors. A simple analogy of the dialogue between designers and software engineers is of two people from different linguistic and cultural backgrounds, processing their thoughts in their native languages and trying to make themselves understood by each other in a third common (but second) language. The difficulties faced by both do not just include that of language and translation, but of the actual understanding of particular words.

Unsurprisingly, the differences in terminology and thought processes between the two disciplines were apparent in the initial work on the validation of the proposed system. Although the proposed design system was not actually built, partially testing the concepts required collaboration between the two disciplines and even at this level, challenges and frustrations were encountered in trying to understand each other. The two

disciplines were ultimately able to overcome these challenges and did succeed in finding a common ground. However, this was only achieved by making an effort to gain a basic understanding of each other's disciplines and finding a common medium in which to communicate; for instance the designer, with a basic understanding of how an algorithm works, was able to make himself understood by the encoders by using a simple pseudo-code.

The challenges faced in this research project serve as a cautionary note for the future development and implementation of the actual system and highlighted the fact that any future development of the proposed design system will require a significant amount of time, effort and cooperation in bringing the two disciplines together. The notion of finding a common ground between design and computer programming is not, in fact, a new one. The subject has been investigated in studies of a new trend in architecture and design education that raises the question of how much mathematics, algorithmic logic and programming architects and designers need to learn (Greenberg, 2007; Marx, 2000; Oxman, 2008; Terzidis, 2006).

The aim is to build the proposed design system with an interface capable of accommodating the design process conducted by both designers, with only limited knowledge in programming. However, some knowledge of algorithms and scripting is expected from the architect-user to produce a useful design schema for the system to process and generate satisfying results.

From a practical prospective, while CAD drafting and the modelling professions do exist specifically for the architecture and design industries, architects and designers themselves do have the knowledge and the capability to perform drafting and modelling tasks. In contrast, computer programming and encoding has not yet been widely specialised for the building industries in a practical sense. Additionally, most architects and designers do not have the knowledge and capability to conduct tasks involved in computational programming (as opposed to computerisation).

Therefore, one can foresee a specialised programming and encoding profession, specifically targeting the building industries, develop out of the CAD drafting and modelling professions, in a similar way to the way that the old drafting profession was replaced with the CAD drafting and modelling profession.

A further challenge that may potentially be faced when building the proposed design system (in the evaluation stage of the overall research project) is the specification of the computer system used. The computational capacity required for the vast number of variants processed at each step of the design system suggests that the required processing power will be substantial. Because the HEAD system has not yet been built, clearly, the minimum computational requirements have not been tested and will only be known when the actual system is built and tested. Having said this, the vast computational requirements of the system as a whole were foreseen and the hierarchical nature of the proposed system is believed to offer a partial solution. The hierarchical arrangement allows the design problem and process to be decomposed so that the system is able to process a fraction of the variants at one time, thereby reducing the required computational capacity. Each level of the system will, nevertheless, be fed a large amount of variants that will need to be processed and thus, irrespective of the hierarchical set-up of the system, it is acknowledged that the computational requirements will be a relevant concern.

6.3 Conclusions

Recent studies suggest that algorithmic design approaches hold the key to taking advantage of both the realm of human creativity and that of computer design systems in the real sense of Computer Aided Architectural Design (CAAD) (Frazer, 1995; Kalay, 2004; Kolarevic, 2003; Kotnik, 2006, 2010; Oxman, 2008; Terzidis, 2006). Algorithmic design approaches are believed to have the potential to mediate between the human mind and the computer. Computer systems in design are developed with the essential computational mechanisms, to name a few, iteration, recursion and the conditional application of rules. Algorithmic computational procedures of

deduction, induction, abstraction, generalisation and structured logic usually automate tiresome labour-intensive routines and demonstrate superiority in quantitative capability over the human mind. Furthermore, algorithms can be developed with an inductive logic to produce unpredictable, unimaginable and inconceivable results, which can be regarded as a computer's creativity analogous to intuition as a source of human creativity (Aranda & Lasch, 2006). Nevertheless, computers are not able to consciously justify simple decisions, because they are lacking in the abstract holistic nature of human thinking (Terzidis, 2006).

While the algorithmic approach brings the computer closer to the human designer, a certain amount of standardisation within the computer-aided design process is still called for. This raises a controversial issue, because standardised design approaches to repeated projects conflict with an essential architectural principle that regards each building as a unique project. It is commonly agreed that the design decisions during the early phase of the design process impose a significant impact on the project's overall construction and operational costs. Clearly, the re-building of successful designs in different contexts other than what they were initially designed for involves a huge risk of failure. The after-the-fact element of attempting to resolve the areas where the building design might be failing to respond would require high-cost solutions. For instance, the building could impose its functional solutions on the users who would have to adapt to it, rather than providing suitable solutions for the targeted users. Also, the building could require many more systems or mechanical loads than it might actually need, if it was designed specifically for its location.

Parametric design introduces a new approach to modularity and offers a way in which the advantages of standardisation in the design process can be gained while maintaining the crucial uniqueness of each specific design. While the basic building blocks and components can be standardised, the design constraints (internal and external) are specific and unique to each design problem. Because the fitness functions that govern the assessment and the selection of the successfully evolved solutions are based on these

constraints, (and therefore, the specific environment and requirements of each design), the ultimate design solution will be unique every time the same standardised building components and processes are used in a different design problem of similar building types.

Building performance simulations are being more widely used to assess the performance of the produced solutions. These building performance simulations serve to highlight areas where improvements can be made and ways in which the solution can be optimised. The results usually highlight many conflicting issues where trade-offs seem inevitable and thus the designer is forced to make subjective design decisions. Much research has been conducted in evolutionary systems in architectural design where it was successfully demonstrated that design solutions can be evolved in response to chosen performance criteria, thereby minimising the need for the designer to make these subjective decisions. However, in the majority of the research carried out, only limited fitness functions were assigned during the evaluation phase and this contradicts the interdisciplinary nature of the building design problem.

This research sought to find a way in which a design schema, of a standardised building design approach, could be formulated and encoded into a generative evolutionary design system, to engender a beneficial relationship between human creativity and computer design applications that can be used to produce interdisciplinary, integrated design solutions that are performant and responsive to the uniqueness of constraints and the environment of each project.

The research was based on the hypothetical proposition that the proposed design method of an integrated interdisciplinary design approach, which acknowledges both the designer's creativity and preconceptions, can be combined with the advantages offered by the algorithmic computation of the generative design systems and the genetic evolutionary systems, to be used from the earliest stages of the design process. The proposed HEAD system connects the evolutionary process directly to the performance

assessment applications as fitness functions organised in order of priority. This HEAD system would facilitate a smooth transition between the designer's creative vision and the computer system by encoding a dynamic design schema into a design system.

Because the research subject is concerned with contributing to the development of a design approach that did not exist, the research adopted a pluralist methodological approach incorporating a number of different methodologies during its cyclical processes. Methodologies suited to the nature of the various research stages were chosen at each stage. The overall research project was developed to be undertaken through the three major stages of 'conceptualisation', 'developmental' and 'evaluation'. This thesis concentrated on the first two research stages and iterations between the two stages were permitted, depending on the understanding of the undergoing design situation.

The proposed design method demonstrates a potential means of digitally formulating the creativity of a human designer into a dynamic design schema that can subsequently be encoded and executed through the use of computational algorithms. Taking into consideration the tectonic design approach, which places a higher aesthetical value on the **process** than on the **product** only, the design schema is flexible and dynamic so that it can be used appropriately for either a unique building design or for a standardised building design. The designer's creativity is translated into a modular design approach of parameterised building blocks. Prolog descriptions of the building's configurations and regulating elements are used to govern and guide the production of the design solutions throughout the permutation and propagation processes.

The very nature of the Hierarchical Evolutionary Design (HEAD) System, which is set-up in a hierarchical, geometrical, multi-level configuration, means that the production of non-viable solutions can be avoided and the computational processes can thus be minimised. Minimising the computational processes required, ultimately means that the time required

for the whole process is greatly cut down. In addition, the hierarchical approach allows the decomposed sub-problems to undergo multiple and sequential assessments using fitness functions. These assessments against fitness functions are carried out throughout the design process and, ultimately, the overall composed design solution is also passed through the assessments that test against the specific fitness functions of the design. Consistency controllers, acting as the base and reference for the design decisions made throughout the evolution process, regulate this holistic approach to the design process.

6.4 Future research

The research achieved its stated aim of producing a theoretical framework for a conceptual design method and the HEAD system. Although the actual building of the HEAD system is not a part of the scope of this research, another research group, working in parallel to this research, has undertaken the actual building of components of the system. The results from the building of the system will be published upon the system's completion. It is incredibly exciting to be given the opportunity to have another research group take the proposed system from the conceptual framework produced by this research and endeavour to demonstrate and implement it into a working design system.

Acknowledging the responsibility of the General Directorate of Military Works (GDMW) to manage almost all of the Saudi Ministry of Defence and Aviation (MODA) projects from their conception to completion, the author, as a member of the GDMW, recognises the advantages of the BIM application and its potential to benefit the organisation. The proposal to incorporate the BIM approach into the GDMW processes has garnered wide interest and acceptance. A plan for the implementation is expected to be initiated in the very near future. Falling in-line with the GDMW shift in direction toward the implementation of the BIM, there is great potential for the practical application of the proposed design method and its design system. Not only is the proposed design method and system aligned to the BIM approach, but the system also has the demonstrated capability of

adopting both the standardised design approach and the uniqueness of each project design, which, for an organisation such as the GDMW, is invaluable.

It is felt that the proposed design method and system and the realm of CAAD in general could benefit from further research into the concept of the design schema as a bridging medium between the human designer and computer design systems. The concept of the design schema holds the potential of being expanded into an even more adaptable means for translating the designer's creativity into a logic-based computer language. Further research needs to be undertaken to develop clearer guidelines and definitions of the design schema, so that any designer can create his/her own schema to suite any project he/she faces. Because the design schema is a process-oriented approach, potentiality it could also be expanded and developed so that it could be used across a number of different design systems.

A final area for further research highlighted by this research project relates to the educational foundations of architects. While CAAD systems are usually built with interfaces that allow designers to operate with no understanding of the embedded computational processes and the operations taking place, the proposed design method and system do require the designer-user to have a certain amount of computation knowledge. As a result of this research, it has become clear that an understanding (if not a thorough in-depth knowledge) of the underlying computational processes will greatly benefit the designer, so that the full potential of the computer can be achieved.

7 Appendices

7.1 Appendix A

Published article

Bukhari, F., Frazer, J. H., & Drogemuller, R. (2010). Evolutionary algorithms for sustainable building design. Paper presented at the *2nd International Conference on Sustainable Architecture and Urban Development*. Amman, Jordan: The Center for the Study of Architecture in the Arab Region (CSAAR). from http://eprints.qut.edu.au/32401/

Evolutionary Algorithms For Sustainable Building Design

Fakhri Bukhari, John Frazer, Robin Drogemuller
Queensland University of Technology, Australia

Abstract

This approach to sustainable design explores the possibility of creating an architectural design process which can iteratively produce optimised and sustainable design solutions. Driven by an evolution process based on genetic algorithms, the system allows the designer to "design the building design generator" rather than to "designs the building". The design concept is abstracted into a digital design schema, which allows transfer of the human creative vision into the rational language of a computer. The schema is then elaborated into the use of genetic algorithms to evolve innovative, performative and sustainable design solutions. The prioritisation of the project's constraints and the subsequent design solutions synthesised during design generation are expected to resolve most of the major conflicts in the evaluation and optimisation phases. Mosques are used as the example building typology to ground the research activity. The spatial organisations of various mosque typologies are graphically represented by adjacency constraints between spaces. Each configuration is represented by a planar graph which is then translated into a non-orthogonal dual graph and fed into the genetic algorithm system with fixed constraints and expected performance criteria set to govern evolution. The resultant *Hierarchical Evolutionary Algorithmic Design System* is developed by linking the evaluation process with environmental assessment tools to rank the candidate designs. The proposed system generates the concept, the seed, and the schema, and has environmental performance as one of the main criteria in driving optimisation.

Keywords: *sustainable design, CAAD, conceptual design, generative design, evolutionary algorithm, genetic algorithm.*

1 A computational architectural design method for generating parameterised sustainable design models

Sustainable designs produced using existing CAD systems might not represent the most sustainable design outcome. This is due to the lack of capacity of current software packages to evaluate any great number of design iterations in a reasonable amount of time.

While Computer Aided Design (CAD) systems have been successfully utilised in architectural design for many years, they have by no means reached their full potential (Boddy, Rezgui, Cooper & Wetherill, 2007). Some scholars refer to the obstacles faced by designers in switching between the conscious thought required to operate a computer and the unconscious flow of creative thought, as the reason that CAD systems are not readily able to adapt to conventional design methods (Liddament, 1999). From this perspective, although computers are being more widely used in the design process itself, it is noticeable that when they are utilised in architectural design, they are most frequently used for drafting and visualisation (Holness, 2006).

Despite the fact that architectural CAD systems have not reached their full potential, there have been promising developments in the field. The three most notable, in terms of their potential to produce sustainable designs, are used as foundation for the proposed design methodology. They are generative design systems, performance based design, and Building Information Modelling (BIM).

The generative design systems considered in this paper generate design alternatives from a simple "seed", according to a set of rules governing the growth process (Frazer, 1974) (Frazer & Connor, 1979). Working in a similar way, but using environmental factors as their basis, are the evolutionary design systems, which evolve designs from a "genetic code" of an architectural concept in response to the environment, through natural selection (Frazer, 1995, p. 65).

Evolutionary systems seek design solutions which have the "symbiotic behaviour" and "metabolic balance" found in the natural environment (Frazer, 1995, p. 9). Essentially, evolutionary systems aim to produce sustainable built environments through a responsive design process. The design itself takes shape as a result of the natural environment, as opposed to conventional design processes where designs are adapted in order for them to respond to the natural environment.

Performance based design, which has come to the forefront of research to a great extent due to the need for sustainability (Kolarevic & Malkawi, 2005, p. 195), is a comprehensive design approach which utilises the computer's capabilities of simulation of targeted performance to generate building forms in response to its simulated environment (Kolarevic & Malkawi, 2005, p. 197). Although qualitative simulation capabilities are still being researched, computers can aid human qualitative assessments through their quantitative capabilities (graphic output, visualisation etc).

Building Information Modelling (BIM) is "a digital representation of the physical and functional characteristics of a facility", (NBIMS, 2007) which can be used as a means for collaboration between all stakeholders throughout the lifecycle of the project. BIM provides an all-encompassing approach to the design of the building, taking into account all aspects of the building's performance simultaneously, which establishes the base from which sustainable designs can be produced.

Project Services, the building procurement arm of the Queensland State Government in Australia, is testing various methods of procurement that exploit the use of BIM (Building Information Model) technology (Holness, 2006). One experiment they conducted involved the thorough analysis of the thermal performance of a range of building forms at the early sketch design stage by mechanical engineers (GBCA, 2009). The architects then developed the optimal design through to contract document stage and construction.

The "standard" process, where normally the architects design the building and only discuss the design with the mechanical engineers once the design has developed was turned around. This enabled the building to achieve the high Green Star rating without relying on unusual techniques to score extra Green Star points. However, this process had a significant cost in human effort to run over 200 analyses using the BIM and thermal analysis software to iterate towards an appropriate solution.

The research being carried out is based on the proposition that the developments in evolutionary systems, performance base design and BIM can be adapted and integrated with each other to create an architectural design process that supports and fosters human creativity in a digital scripted form. This paper presents a method to automate much of this process. This would eventually allow the exhaustive analysis of options to be more widely used.

It is expected that a generative formation design model can be developed that simulates the project's environments and also simulates its desired performance to generate sustainable design solutions in response to the selected environmental parameters. Thus, the conceivable possibilities will be assessed and the outcomes will be objective measurements of the best choices made through optimisation.

The proposed design system operates during the pre-design and schematic design phases of the design process. Here, the designer is able to make informed and objective decisions which will have a higher impact on the quality of the design and a lower impact on cost. "To produce smart sustainable designs within [man hour and time constraints] designers need to be disciplined and focus efforts more at the early schematic design stage, and to test options while there are fewer constraints on the process." (Kolarevic & Malkawi, 2005, p. 45).

The proposed system, illustrated in Figure 1, seeks sustainable design solutions during the earliest and most critical phases of the design process.

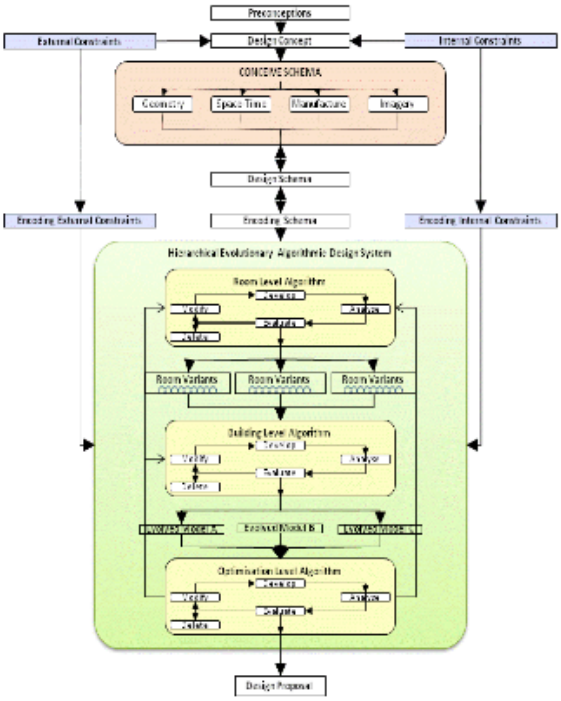


Figure 1 An illustration of the proposed design method.

The results presented in this paper have been fully specified as a process and worked through as "paper studies" to ensure that the algorithms and design factors are appropriate. The system is currently being implemented in computer software.

1.1 Design Context

The design context incorporates every aspect of the design problem, which is very often not immediately obvious but comes to light through the problem solving process (Lawson 2006). As most design problems seek to satisfy multiple functions which are both overlapping and interacting, design solutions need to respond to a multitude of requirements. The generators (client, user, designer and legislator) of the design constraints, both internal and external, seek to respond to four types of functions; formal, symbolic, radical and practical (Lawson, 2006).

Bertel, Freksa and Vrachliotis (2004) and Lawson (2006), describe constraints as the "result from required or desired relationships between two or more elements" of the design problem (Bertel et al., 2004, p. 267).

1.1.1 Design constraints

Two types of constraints were recognised; 'External Constraints' and 'Internal Constraints'. The external constraints are the unchangeable part of the design problem which are not affected by the design method and cannot be changed without creating a further problem. For instance, site (location, orientation and neighbouring built environment), weather, building regulations and the laws of physics are all prime examples. The internal constraints such as functional expression or modes of fabrication, on the other hand, have some level of flexibility and can be changed without creating problems. These values vary through the design process and are determined by the designer.

The external constraints simulate the site's external physical environment with its unchangeable (within the scope of the project) facts. The internal constraints animate the hosted activities in ideal and extreme conditions to virtualise the targeted internal environment of the building.

Bertel et al. (2004, p. 267) cited Carrara, Yehunda and Novembri (1994) to suggest that "a prioritisation of goals, reflecting a descending order of preferences, may be imposed by the designer or by the client, [indicating] which combination of performance criteria the designer should attempt to accomplish first". The prioritisation of the project's constraints to meet with the project's goals is highly influenced by the designer's preconceptions.

1.1.2 Designers' preconceptions

A number of researchers argue that architects have preconceptions as they design. Broadbent modified the 1960s design process by including 'preconceptions' in the synthesis stage of the process (Broadbent, 1988). These preconceptions play an essential role in the creative aspects of the design process (Janssen, 2004). Two types of preconceptions can be identified;

Guiding Principles - Variously called the designer's 'paradigmatic stance' (Broadbent, 1988), 'guiding principles' (Lawson, 2006) and 'theoretical position' (Rowe, 1998). The guiding principles reflect the philosophical beliefs,

cultural values and background of the designer. This stance is evolved and developed all the way through the designer's practicing career (Janssen, 2004). The guiding principles contribute to the designer's preconceptions by influencing the capture of the design concept.

Primary Generators - Various scholars attribute the formation of these initial ideas to a variety of sources; the 'primary generators' (Drake, 1979), 'enabling prejudices' (Rowe, 1998), 'concept or parti' (Lawson, 2006) and 'Generative Concept' (Frazer, 1974) (Frazer & Connor, 1979) (Frazer, 1995). The primary generator influences the conception of the design schema.

The guiding principles of the designer's preconceptions influence the prioritisation of the design constraints according to the project's goals and objectives. The subjectivity in the prioritisation of the design constraints influences the capture of the design concept based again on the designer's guiding principles (Frazer, 2002). Abstracting the captured design concept, the designer's primary generator influences conceiving the design schema in reflection to the third type of constraint in Lawson's (2006) observations, which is derived from the designer's stance and is thus a 'self-imposed' constraint (Janssen, 2004).

1.2 Design Schema

A design schema is a highly customisable but generic working method. As an adaptable design model inserted into the design process, the design schema characterises the designer's style as an abstract conception of common features of their designs (Janssen, Frazer & Tang, 2002). The design schema is also employed to maintain the designer's creative expression as a central part of the design process. The design schema, allows transfer of the human creative vision into the rational language of a computer. The design schema leaves the door open for the continuance of human creativity to invent new design tools.

The design schema translates the design concept with a digitally parameterised representation and a set of defined transformation rules. The design schema is then elaborated into the use of genetic algorithms to evolve innovative and performative design solutions.

1.2.1 Conceiving the design schema

Several scholars have studied design principles, from the conventional paper based architectural design practice and its descended digital version through to the digital architectural design practice.

Rowe (1998) presented five forms of heuristics that a designer may use in the search for a design solution; (i) 'Anthropometric Analogies' – which use the human form as an inspiration, (ii) 'Literal Analogies' – which use established and readily recognised shapes and forms, (iii) 'Environmental Relations' – which relate to the building's performance within its environment, (iv) 'Typologies' – using established design solutions and (v) 'Formal Language' – which uses the

formal, accepted solution as a guide. Kolarevic (2003) introduced six methods of digital morphogenesis, (i) Parametric Design (ii) Dynamic and Field of Force (iii) Datascape (iv) Metamorphosis (v) Genetics (vi) Performative. Flanagan (2005) presented four generative methods in digital design, (i) 'Generative Geometry' (ii) 'Generative Imagery' (iii) 'Generative Manufacture' (iv) 'Generative Space-time'.

Without having to make an in-depth comparison of the above three approaches, it can be seen that they share some common elements. Which of these fundamentals is used is based on the designer's preconceptions but any one of them can nevertheless be used to interpret the design concept in a digital representation.

1.2.2 Encoding the design schema

The design schema can be encoded (Janssen, 2004) in one of three ways. The highly-generic approach applies standard rules and representations using binary strings which do not rely on any domain or task-specific knowledge. Performance of an evolutionary algorithm in generating designs, based on broad generalisations, is not necessarily highly efficient. However the lack of performance is compensated by re-usability. The domain-specific approach incorporates domain-specific knowledge in the rules and representations to improve the performance of the system, however there is a corresponding loss in re-usability. Finally, the task-specific approach can be used when the type of task in the domain is complex and a high level of performance is required. The task-specific approach is at the opposite end of the spectrum to the generic approach. While it offers the highest levels of performance, re-usability is very low.

In the domain of building designs, when the task-specific rules and representations of a specific designer are encoded, the system will reflect the designer's preconceptions and style. Therefore, the domain-specific approach is found to evolve more generic designs and it is also more flexible for re-use by other designers. In addition, the domain-specific approach evolves designs with high variability but with poor efficiency. On the other hand, the task-specific approach compensates the re-usability and style issues by evolving more surprising and challenging designs with high efficiency.

1.3 Generation and Formation

Nature, ever changing and evolving, serves as a rich source of inspiration for designers and researchers attempting to optimise the performance of their designs. This is not a new concept, John Frazer coined the phrase 'Evolutionary Architecture' (1974) (1979) (1995), referring to the investigation of, and search for form-generating processes in architecture, based on the notion of morphogenesis in the natural world. Frazer's conceptualisation of architecture goes beyond the idea of shapes and forms to encompass a set of rules that generate spaces and forms. The built environment is thus produced through a

series of evolutionary steps based on the selection of solutions that best respond to the fitness function criteria.

Genetic algorithms are used to simulate the concept of evolution with computer logic. Genetic Algorithms were first presented in the 1960s by John Holland in his investigation of the process of natural selection systems. As described by Holland (1992), the structure of genetics consists of a population of chromosomes which represent possible solutions to a problem. Genetic operators, such as crossover and mutation between two or more genotypes produce a new generation phenotypes. The genetic make-up of the next generation is dependent on the process of selection which decides on what part of the population will pass to the next generation. Every solution in the population is evaluated and selection is made on the basis of fitness. As genetic algorithms deal with a huge number of possible solutions, as opposed to just one solution at a time, it is much more likely that an acceptable solution will be found (Kalay, 2004, p. 283).

1.3.1 A responsive generative formation design model

The concept of the designer as a tool maker (Janssen et al., 2002) is introduced here as the simulation of the project's environments are employed as a tool to aid in the generation of design solutions.

Simulated External and Internal Environments; The Evolutionary Design Systems, on the one hand, using a repeated cycle of genetic scripted code, are able to produce designs adapted to the environment. Performance Based design on the other hand, applies modifications to the produced design, based on its performance in an analytical simulation. The proposed design method develops a generative formation design system with two environments encoded into the system. The project's external constraints simulate the external environment of the project. The project's internal constraints simulate the internal environment and animate the desired performance of the project. Thus, the design would be generated in response to the simulated external and internal environments and also to the animated desired performance.

Prioritisation of constraints; all the project's identified constraints are encoded according to the relative priority assigned to them by the designer. The generative formation system will act in favour of the constraint which is assigned the highest priority. For example, if the building orientation is determined to be more important than environment, the environmental solution will be applied on the best orientation solution. On the other hand, if the environmental constraint is set with a priority over that of orientation, the best environmental setting determines orientation. Thus, each prioritisation produces a different generation of solutions with a set of different characteristics.

The prioritisation of the project's constraints and the subsequent design solutions synthesised in accordance with the simulation of the project environment during design generation is expected to resolve most of the major conflicts, in the evaluation and optimisation phases. The system produces a

responsive design solution, optimised and developed through its creation process in simulated and animated environments, thus responsive to its living environment, responsive to the captured performance and responsive to the designer's creativity which is reflected in the design schema.

1.3.2 Hierarchical Evolutionary Algorithmic Design System

The system is designed to incorporate the notion that the design problem cannot be "comprehensively stated", "require[s] subjective interpretation" and "tend[s] to be organised hierarchically" (Lawson, 2006, p. 120). The system is divided into three levels to respond to the hierarchical decomposition of the design problem into further levels of sub-problems. Each level of the HEAD system is thus able to tackle issues of sustainability sequentially.

The first level, *the room level algorithm*, generates successful variants of possible solutions for each required space based on assigned fitness functions. The fitness functions include the project's design criteria, design standards, building codes and functional requirements applicable to individual rooms. The evolved successful generations are available for the second level of the system.

The second level, *the building level algorithm*, combines one variant of each generated space from the first level according to the adjacency requirements, mode of fabrication and a grid system. The successful configurations of the space lay-out are then fed into the third level.

The third level, *the optimisation level algorithm*, optimises the successfully evolved building configurations based on over-all thermal, light, acoustics and cost performance.

There are no optimal solutions to any design problem but rather an infinite number of possible solutions (Lawson, 2006). The design process is never-ending and designers work in the context of a need for action (Lawson, 2006). Thus the system loops back to earlier levels allowing the system itself or the designer to make modifications on the successful selected solution.

1.4 Optimisations

Performance analysis on the chosen model can be conducted by *the optimisation level algorithm* to verify different performance aspects of the produced design. Although available performance analysis applications are valuable tools, they do not normally come with modification capabilities. To overcome this issue when modifications are required or even to test the impact of design decisions, alternative design solutions can be fed back into an earlier phase by varying chosen parameters.

Unlike the performance-based design case where modifications are done after synthesis by evaluating the performance of the produced design based on analytical simulation, in our case, the generation and formation of the design happens in response to the synthesis simulation of the project's environments. This means that the system will only produce solutions from the design space of

viable designs for the design situation without having to generate all design situations that could possibly be produced by every combination.

Thus, the system will have fewer design solutions to produce and to run through the optimisation cycle in order to identify the successful model(s). This also means that the modification capability could be added to the optimisation cycle within the generative system to enable automatic adjustments and development.

2 Research Framework

2.1 Plan

To test the proposed hypothetical proposition the intention is to build a system specifically to design a particular building typology. It is not intended to build the responsive generative formation design system in its full capacity. At this stage a test system will be built around the specific pilot project.

A mosque is selected as a pilot project. Mosques have more defined design problems due to the fact that specific planning and design criteria arise from a set of explicit religious requirements. These distinguishing features would give greater control over design variants and thus a more controlled design space and generation of solutions. On top of that, these features provide a reference to the functional evaluation of the produced design solution.

The system will be set up to simulate the selected pilot project's environments and its desired performance based on the prioritised external and internal constraints of a selected existing mosque project.

2.2 Action

A retrospective BIM performance analysis will be carried out on the selected mosque to establish a base against which the design solutions produced by the proposed model can be compared. The intuition of selecting a mosque design as the pilot project arises from the need to maintain control over design variants while regenerating different design solutions for different site conditions. Two scenarios are proposed:

- Design a new schema for the conventionally designed selected mosque and process it through the Hierarchical Evolutionary Algorithmic Design System to generate a responsive design solution. The design solution will capture the codified concept abstracted into the design schema in response to its environment.
- Run the design schema from above through the Hierarchical Evolutionary Algorithmic Design System, but at a different geographical location, to generate a responsive design solution to the new environment.

2.3 Evaluate

The design solutions produced by both of the scenarios discussed above, as well as the conventionally produced design solution of the pilot project, will be evaluated according to environmental performance analysis, functional performance analysis, cost estimates and all other associated fitness functions.

The interoperable nature of the produced design models will allow immediate performance analysis to be carried out to verify various performance aspects of the design.

3 Pilot project: Mosques

3.1 Mosque Architecture

Mosque architecture has been a controversial issue for as long as mosques have existed. One school of thought abandons the charged ornaments all over the building and seeks simplicity in design based on religious functionality, much like those built during early Islam. The other school of thought gives higher aesthetic and social value to the place, acknowledging the Muslim architectural heritage and emphasizing symbolic features that have become the stereotypical image of the mosque.

Mosque typologies have developed over the centuries, benefiting from the diverse and dispersed cultures all over the world while maintaining their identity (Frishman, Khan & Al-Asad, 1994). Each of the Muslim ecological cultural regions has contributed to mosque architecture. With defined spatial organization principles and a set of generic forms, historic mosques reflected the 'eclectic and integrative' nature of Islamic architecture (Ardalan, 1980).

Nader Ardalan (1980) presented a preliminary survey covering 113 mosques around the world in which he identified a typology of spatial organization and distinguishing generic Islamic forms. The Islamic world was categorized into six regions based on ecological and cultural backgrounds. Each zone is presented with a dominant typology; The Arabian Peninsula and North Africa; dominated by the Hypostyle with Dome Accent typology. Turkey; dominated by the Central Dome typology. Central Asia; dominated by the Four Iwans typology. India; dominated by the Hypostyle with Domical Vaulting typology. East and West Africa; dominated by the Hypostyle typology. Far east; dominated by the Complex typology.

Although a dominant typology found in each zone is clearly noticeable, other typologies are also found in the same region. For example, in the zone of the Arabian Peninsula and North Africa all typologies were found. Despite the fact that the 113 samples were not equally divided between the six zones, eight generic forms were found in each zone and typology; the niche *'Mihrab'*, the courtyard, the dome, the Minaret, the gateway, the portico, the ablution place and the plinth.

Looking at the spatial categories of the historic Mosques in Ardalan's research, we find that most mosques consist of two main parts; the prayer hall and the courtyard. The prayer hall includes the Mihrab, gateways and a portico. The courtyard also includes gateways, minaret(s), an ablution place and a portico. Additional spaces for required functions are added to either part accordingly. All the other spaces and generic forms are hosted and represented within these two parts.

Ardalan (1980) noted some spatial organization principles; direction, introversion, centrality, symmetry and symbolic. The main spatial organization principle is the orientation to Makkah. The Mihrab in the centre of the Qiblah wall is the architectural organic form found in each and every mosque since the first mosque in Madinah was built 1430 years ago to orientate the prayers toward the direction of the Kaabah in Makkah. Introversion, which comes from the prayer ritual itself, is served with the courtyard and central dome planning. The gateway and the portico, as transitional spaces, buffer the prayers from distractions. Centrality and symmetry, which also come from the prayer rituals during the formation of the rows behind the Imam on both sides equally, are reflected in the domical and the pyramidal forms highlighting the main sacred space. The principle of symbolic spatial organisation is seen in the use of the plinth, especially in a single plan courtyard, which adds the value of a raised space to the mosque.

Environment has a great influence on mosque planning and architecture. For example, the courtyard is designed to provide, among other functions, natural lighting and fresh air to the prayer hall. The courtyard of the mosques found in Arabia, Persia and India are spacious, where as those found in Turkey and central Asia are smaller, and in some extremely cold regions have completely disappeared. The porticos are designed to provide shaded areas surround the courtyard of most of the mosques built in Egypt and the eastern Muslim countries. In the western Muslims countries and Andalusia, the porticos where covered for protection against heavy rain.

Modern mosques follow the spatial organization principles of historic mosques and keep the essential generic forms, developing in response to economical, cultural and environmental issues. Mosque typologies which emerged from the ecological cultural regions have, over time, become symbolic. Emphasis of some generic forms such as the dome and the Iwans are noticeably vanishing in modern mosques. Advancements in technology, building systems and building materials are greatly influencing modern mosque design.

However, Aksamija (2007) cites Rafael Moneo as stating that, "the work of Architecture is irreducible to any classification", suggesting that typology is based on the continuous development of shared characteristics (Aksamija, 2007), which always result in a fitter generation of types. Looking at mosques from a genetic development perspective and based on Moneo's concept of developing types, we can note that mosque typology has indeed developed from a gene pool of generic forms which follows a typology specific to ecological regions. For

instance the first mosques were built as simple hypostyle prayer halls. The Mihrab, the pulpit *'Minbar'*, the minaret and the courtyard were introduced later, followed by other generic forms which were influenced by the various ecological Muslim regions. The emphasis of a certain generic form within the combination of generic forms, with respect to the principles of spatial organisation, came about in answer to the specific needs and architectural language of a particular region. Mosque architectural development can thus be looked at from a genetic perspective where certain generic forms are introduced by an ecological region as chromosomes, resulting in a distinguishable typology; for example the dome, the Iwans and the pyramidal roof were all introduced as genotypes which resulted in a phenotype typology. Regional genes or generic forms are adopted by other regions, fusing into another generation of typologies.

Looking at the mosque from the perspective of developing typologies, we can identify six regions with six dominant typologies. These typologies are all influenced by their environment and culture, resulting in a set of guidelines or a set of shared characteristics specific to each typology. The characteristics of each typology can be formalised by different schemas, where each schema produces several successful permutations.

3.2 Mosque design schema

The design schema for the pilot project is inspired by a combination of selected elements from Rowe, Kolarevic and Flanagan's approaches. The schema applies Rowe's *lateral analogies*, *environmental relations*, *typology* and *formal language*, with Kolarevic's *parametric design*, *genetics* and *performative morphogeneses*, as well as Flanagan's *generative geometry* and *generative manufacture*. The design schema is conceived on a task-specific basis, that is to say, it is specifically designed for a mosque design project

Through observation and with reference to Rowe's *lateral analogies*, *typology* and *formal language*, we note that all the generic forms are included within two main parts of the mosque; the prayer hall and the courtyard. As a result we can distinguish mosques that have only a prayer hall, hosting the generic forms associated with it and mosques that combine the prayer hall with a courtyard, which include the generic forms associated with both.

Each generic form and space is parameterised according to the *room level algorithm's* assigned fitness functions. The genetic algorithm explores all variants and evolves successful permutations.

A Prolog programme using definite clause grammar (DCG) is developed to build seeds from high level descriptions of various mosques, resulting in a symbolic representation of sequence strings out of the different configurations, to govern *the building level algorithm*.

The spatial organisations are graphically represented according to the adjacency constraints between spaces. This form of representation is not new (Levin, 1964) but is still being proposed to support analysis (Wu, Lee, Koh,

Aouad & Fu, 2004). Each configuration is represented by a graph where nodes are placed to represent spaces which are linked by relationship lines analogous with graph and circuit theories. The adjacency data in the graph is processed as either true or false, which does not allow the priority to be defined. The use of weighted adjacency matrices could generate different graphs.

Each graph is then translated into non orthogonal duals (Grason, 1970) and fed into the genetic algorithm system and expected performance criteria are set to govern evolution in the *optimisation level algorithm*. Each generic form and room is parameterised with functional and performance criteria such as lighting, thermal and acoustic performance used as fitness functions during assessment and selection.

Within the context of mosque design, the orientation towards Makkah for the prayer hall, and the module of the prayer mat within the prayer hall, are the two overriding design constraints. The courtyard and ancillary spaces can be oriented in whichever direction is appropriate. The fitness functions that are used to select the "best" members of the population generated by the system are heat flow for the four solstice and equinox dates, cost as a function of the sizes of the various building components and CO_2 equivalent emissions of the building materials.

4 Conclusion

Performance evaluation systems available today are not equipped to evaluate the large number of design iterations which need to be assessed in order to find the most sustainable design solutions. In addition, these systems do not have modification capabilities. Using performance-based design, modifications are done after synthesis by evaluating the performance of the produced design based on analytical simulation.

Bringing together the evolutionary systems, performance based design and BIM system, the proposed HEAD system aims to produce sustainable designs through a responsive design process. Sustainable performance targets, whether environmental, economic or social, can be set for any or all aspect of the design, and design solutions are generated in response to these simulated targets. The generated design solutions can then be tested using BIM, allowing for sustainable choices to be made early in the design process.

The research explores the level of human interventions during the early design process in a guided evolutionary environment, aiming for a design system that facilitates multidisciplinary integration in the early stages of the design process. Environmental performance is one of the main criteria to drive optimization by linking the evaluation process of evolutionary algorithms with environmental assessment applications to rank the candidate designs. In the case of the proposed HEAD system, the generation and formation of the design happens during the synthesis stage, in response to the simulation of the project's targeted environments.

As Mahadev Raman points out "professionals tout definitions of sustainability but the practical application of these concepts presents major challenges" (Kolarevic & Malkawi, 2005, p. 43). If sustainability is taken to be "about finding the right balance between environmental, economic and social concerns" (Kolarevic & Malkawi, 2005, p. 43) then the proposed system holds the potential of producing sustainable design solutions.

5 References:

Aksamija, A. (2007). The Generic Mosque. Design Principles for Mosque Design. *Critical Design* (24), 51-61. Retrieved 22/04/2009, from http://tdd.elisava.net/coleccion/24/aksamija-en/view?set_language=en

Ardalan, N. (1980). The Visual Language of Symbolic Form: A Preliminary Study of Mosque Architecture. *Architectural Transformations in the Islamic World* Fez, Morocco: Aga Khan Award for Architecture.

Bertel, S., Freksa, C., & Vrachliotis, G. (2004). Aspectualize and Conquer in Architectural Design. In J. Gero, B. Tversky & T. Knight (Eds.), *Visual and Spatial Reasoning in Design III* (pp. 255 – 279): Key Centre of Design Computing and Cognition, University of Sydney

Boddy, S., Rezgui, Y., Cooper, G., & Wetherill, M. (2007). Computer integrated construction: A review and proposals for future direction. *Advances in Engineering Software, 38*(10), 677-687.

Broadbent, G. (1988). *Design in architecture: architecture and the human sciences*. London, UK: David Fulton Publishers Ltd. (Original work published 1973 John Wiley & Sons, New York, USA).

Drake, J. (1979). The Primary Generator and the Design Process. *Design Studies, 1*(1), 36-44.

Flanagan, R. H. (2005). Generative logic in digital design. *Automation in Construction, 14*(2), 241-251.

Frazer, J. (1995). *An Evolutionary Architecture*. London, UK: Architectural Association Publications.

Frazer, J. (2002). Creative design and the generative evolutionary paradigm. In D. Corne & P. Bentley (Eds.), *Creative evolutionary systems* (pp. 253-274). California, San Francisco: Morgan Kaufmann Publishers Inc. (Elsevier).

Frazer, J. H. (1974). Reptiles. *Architectural Design*, 231-239.

Frazer, J. H., & Connor, J. M. (1979). A Conceptual Seeding Technique for Architectural Design. *Proceedings of International Conference on the Application of Computers in Architectural Design* (pp. 425-434). Berlin: Online Conferences with AMK.

Frishman, M., Khan, H.-U., & Al-Asad, M. (1994). *The mosque: history, architectural development & regional diversity*. New York: Thames and Hudson; c1994.

GBCA (2009). Brisbane home to Australia's greenest building. Retrieved 11 May 2010, from http://www.gbca.org.au/media-centre/news/brisbane-home-to-australias-greenest-building/2642.htm

Grason, J. (1970). A dual linear graph representation for space-filling location problems of the floor plan type. In G. T. Moore (Ed.), *Emerging Methods in Environmental Design and Planning* (pp. 170-178). Cambridge: MIT Press.

Holland, J. H. (1992). *Adaptation in natural and artificial systems : an introductory analysis with applications to biology, control, and artificial intelligence* (1st ed.). Cambridge, Mass: MIT Press.

Holness, G. V. R. (2006). Building Information Modeling. *ASHRAE Journal, 48*(8), 38-46.

Janssen, P., Frazer, J., & Tang, M.-x. (2002). Evolutionary Design Systems and Generative Processes. *The International Journal of Artificial Intelligence, Neural Networks, and Complex Problem Solving Technologies, 16*(2), 119-128.

Janssen, P. H. T. (2004). *A design method and computational architecture for generating and evolving building designs*. Unpublished Ph.D., Hong Kong Polytechnic University (People's Republic of China), Peoples Republic of China. Retrieved from ProQuest Information and Learning Company database.

Kalay, Y. E. (2004). *Architecture's new media: principles, theories, and methods of computer-aided design*. Cambridge, Mass.: MIT Press.

Kolarevic, B. (2003). *Architecture in the digital age: design and manufacturing*. New York ; London: Spon Press.

Kolarevic, B., & Malkawi, A. (2005). *Performative architecture: beyond instrumentality*. New York: Spon Press.

Lawson, B. (2006). *How Designers Think: The Design Process Demystified* (4 ed.). Burlington, MA: Elsevier. (Original work published 1980).

Levin, P. H. (1964). Use of graph to decide the optimum layout of buildings. *Architect Journal, 140*, 809-815.

Liddament, T. (1999). The computationalist paradigm in design research. *Design Studies, 20*(1), 41-56.

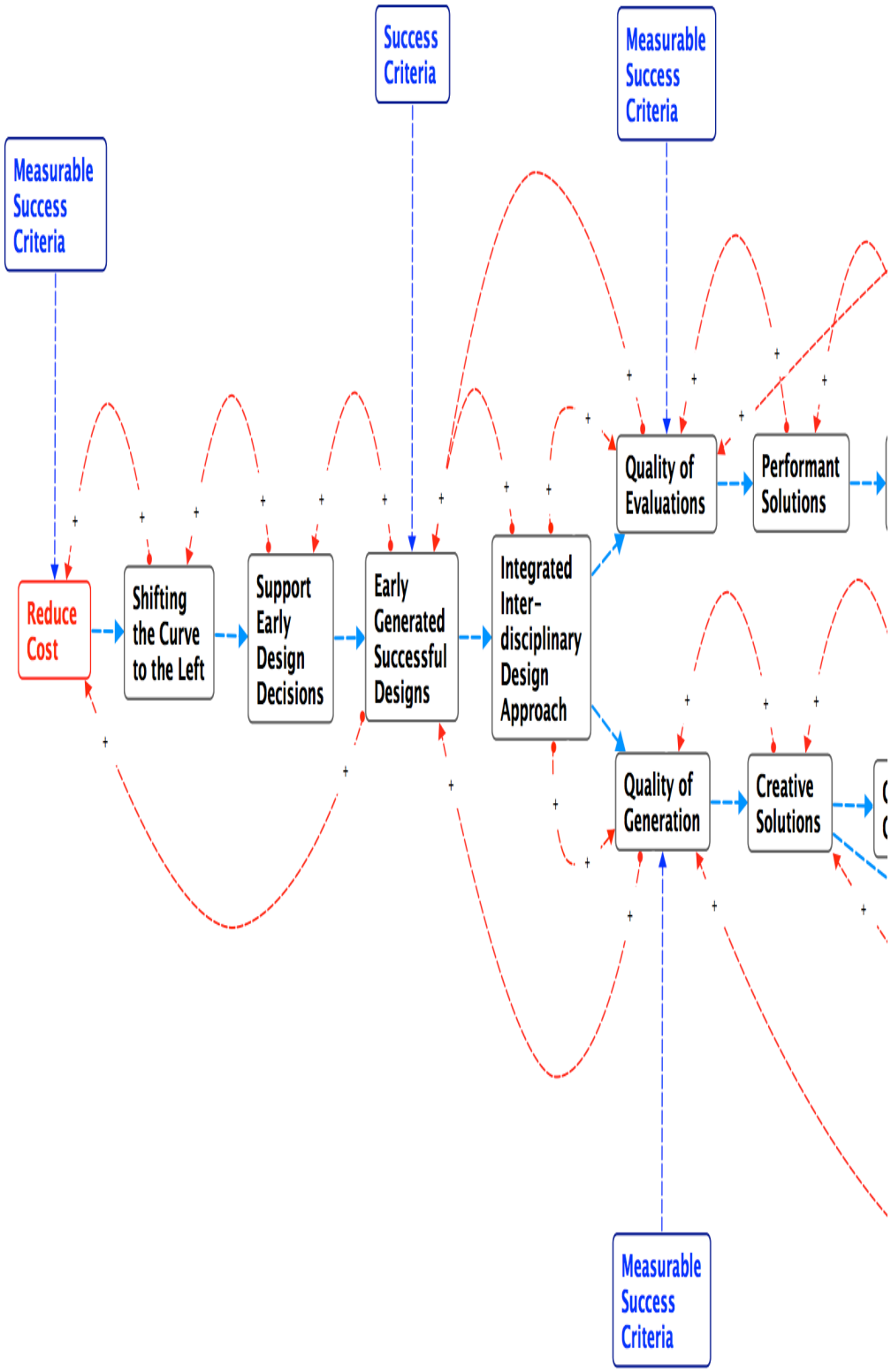
NBIMS (2007). National Building Information Modeling Standard (NBIMS)™ : Overview, Principles, and Methodologies. In N. I. o. B. Sciences (Ed.) (Vol. Part 1, pp. 183): NIBS.

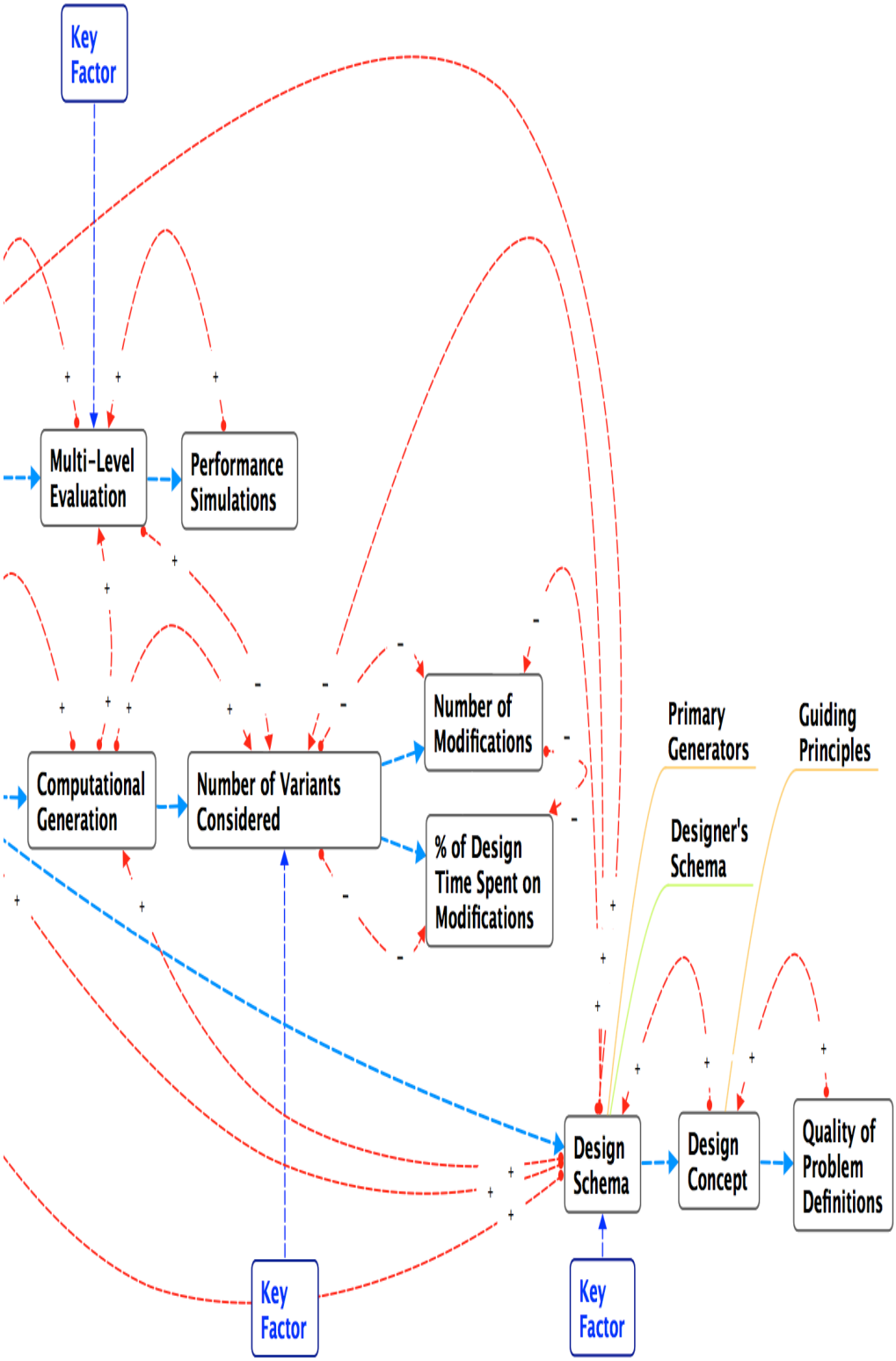
Rowe, P. G. (1998). *Design thinking*. Cambridge, Mass.: MIT Press. (Original work published 1987).

Wu, S., Lee, A., Koh, W. W. I., Aouad, G., & Fu, C. (2004). An IFC-based space analysis for building accessibility layout for all users. *Construction Innovation, 4*, 129-141.

7.2 Appendix B

The developed reference and impact model, with key factors, successful criteria and the measurable successful criteria (next two pages).


Measurable Success Criteria
Success Criteria
Measurable Success Criteria
Reduce Cost
Shifting the Curve to the Left
Support Early Design Decisions
Early Generated Successful Designs
Integrated Inter-disciplinary Design Approach
Quality of Evaluations
Performant Solutions
Quality of Generation
Creative Solutions
Measurable Success Criteria


Key Factor
Multi-Level Evaluation
Performance Simulations
Computational Generation
Number of Variants Considered
Number of Modifications
% of Design Time Spent on Modifications
Primary Generators
Guiding Principles
Designer's Schema
Design Schema
Design Concept
Quality of Problem Definitions
Key Factor
Key Factor

7.3 Appendix C

7.3.1 Mosque Architecture

Since the birth of Islam, over the last one and a half millennia, mosque architecture has become a controversial issue. One school of thought abandons the charged ornamentation all over the building and seeks simplicity in design based on religious functionality. Mosques built during early Islam were built with this functional simplicity and it is believed by some that all mosques should seek to emulate the mosques of the founders of Islam. The other school of thought places a higher value on the aesthetic and social considerations, acknowledging Muslim architectural heritage and emphasizing the symbolic features that have become the stereotypical or iconic image of the Mosque.

Islam, the third of the monotheistic Abrahamic faiths, is the dominant religion in over fifty countries (2010) and the results of many studies suggest that Islam is one of the fastest-growing religions around the world (Allam, 2002; Frishman et al., 1994; Jenkins, 2006). Muslims are estimated as making up over 21% of the world's population, according to the American Central Intelligence Agency (Central Intelligence Agency., 2010). Given these statistics, it is probable that a great many mosques are (and will continue to be) built around the world and the importance of the design of mosques is thus highlighted.

Prayer is the second tenet of Islam's five pillars (or duties) that every Muslim is bound by. The five pillars are the testimony of faith (*shahada*), performing regular prayers (*salah*), giving alms (*zakah*), fasting during the month of Ramadan (*sawm*), and the pilgrimage to Makkah when capable (*hajj*) (Frishman et al., 1994; Ibn Kathir, 2003). The Islamic place of worship is the Mosque, which is an English word from the French (Mosque'e) derived from the Spanish word (Mezquita), which, in its turn, was derived from the Arabic word (*Masjid*) (Haji Mohamad Rasdi, 1998). The root of the Arabic word (*Masjid*) comes from the word (*sajad*), which means to prostrate. The act of prostration takes place several times in each

of the prayer rituals performed by a Muslim five times every day. The word *Masjid* in Arabic is used, in a general sense, to refer to any place of prostration, but its more specific usage refers to the Muslim place of worship, much like the English word Mosque.

The first mosque in the world was the Kaa'ba in Makkah, which was built before Islam by Abraham, with the assistance of his son Ismail, upon an order from God, as mentioned in verse 127 Chapter 1 and verse 96 Chapter 3 of the Qur'an (Ardalan, 1980; Hilali & Khan, 2002, pp. 36, 96; Saheeh International, 1997, pp. 24, 78). The oldest mosque built in Islamic times is the (*Quba*) Mosque in Al-Madina, mentioned in verse 108 Chapter 9 of the Qur'an (Hilali & Khan, 2002, p. 267; Ibn Kathir, 2003; Saheeh International, 1997, p. 264), which was built by the Prophet Muhammad (peace be upon him) and his companions when he immigrated to Al-Madina in the year AD 622. Mosques symbolise Islamic monotheism and the unity of the Muslim community. Over the millennia, the functions of the Mosque have extended from being a place of worship to a social, educational, administrative and political centre (Haji Mohamad Rasdi, 1998).

All the mosques that were built during the time of the Prophet Muhammad (peace be upon him) and the first four 'Rightly Guided Caliphs' embody simplicity and humbleness (Haji Mohamad Rasdi, 1998) in recognition of one of the main principles of Islam, which opposes giving a sacred status to worldly material things. Over the generations however, mosques gradually started to reflect the political power of the Caliph (spiritual head of state) and his reign (Frishman et al., 1994).

Mosque typologies have developed over the centuries, benefiting from the diverse and dispersed cultures all over the world, while maintaining their identity (Frishman et al., 1994). Each of the Muslim ecological cultural regions has contributed to the Mosque architecture. With defined spatial organization principles and a set of generic forms, historic mosques reflected the 'eclectic and integrative' nature of Islamic architecture (Ardalan, 1980).

Nader Ardalan (1980) presents a preliminary survey covering 113 Mosques around the world in which he identifies a typology of spatial organization and distinguishes generic Islamic forms. He categorises the Islamic world into six regions based on their ecological and cultural backgrounds. Each zone is presented with a dominant typology. The Arabian Peninsula and North Africa are dominated by the Hypostyle with Dome Accent typology. Turkey is dominated by the Central Dome typology. Central Asia is dominated by the Four Iwans typology. India is dominated by the Hypostyle with Domical Vaulting typology. East and West Africa are dominated by the Hypostyle typology. And finally, the Far East is dominated by the Complex typology.

Although a dominant typology is found to be clearly discernable in each zone, other typologies are also found in the same region. For example, in the Arabian Peninsula and North Africa zone, all typologies were found to exist. Despite the fact that the 113 samples in Ardalan's survey were not equally divided between the six zones, eight generic forms were found in each zone and typology — the niche '*Mihrab*', the courtyard, the dome, the Minaret, the gateway, the portico, the ablution place and the plinth (Ardalan, 1980).

Examining the spatial categories of the historic mosques in Ardalan's research, we find that most mosques consist of two main parts, the prayer hall and the courtyard. The prayer hall includes the *Mihrab*, gateways and a portico. The courtyard also includes gateways, minaret(s), an ablution place and a portico. Additional spaces for any other required functions are added accordingly to either part. All the other spaces and generic forms within the mosque are hosted and represented within these two parts.

Ardalan (1980) notes some spatial organization principles, direction, introversion, centrality, symmetry and symbolic. The main spatial organization principle is the orientation to Makkah. The *Mihrab* in the centre of the *Qiblah* wall is the architectural, organic form found in each and every mosque since the first mosque in Madinah was built 1430 years

ago to orientate the prayers toward the direction of the *Kaa'ba* in Makkah. Introversion, which comes from the prayer ritual itself, is served by the courtyard and central dome planning. The gateway and the portico, as transitional spaces, buffer the prayers from distractions. Centrality and symmetry, which also come from the prayer rituals through the formation of the rows behind the Imam on both sides equally, are reflected in the domical and the pyramidal forms highlighting the main sacred space. The principle of symbolic spatial organisation is seen in the use of the plinth, especially in a single plan courtyard, which adds the value of a raised space to the mosque.

On the one hand, many modern mosques adopt the concept of the Mosque as a 'House of God' to symbolize the glory of Islam (Haji Mohamad Rasdi, 1998), following the spatial organization principles of historic mosques and magnifying the essential generic forms. On the other hand however, mosque typologies that emerged from the ecological cultural regions have, over time, become symbolic when adopting the new concept that calls for the Mosque to regain its status as a community development centre (Haji Mohamad Rasdi, 1998). The emphasis of some generic forms, such as the dome and the Iwans, are noticeably vanishing in modern mosques. Advancements in technology, building systems and building materials are greatly influencing modern mosque design, which is developing in response to economical, cultural and environmental issues.

7.3.2 Mosques design criteria: general

The following is an outline of the general design criteria of mosque design and planning collected and summarised by the author from the following resources, (i) a printed manual by Hazem Mohamed Ibrahim (1979) titled *'Planning Standards for Mosques'* compiled for the Ministry of Municipal and Rural Affairs in Saudi Arabia, jointly with the Development Program of the United Nations, (ii) a report by the Value Engineering Section of for the General Directorate of Military Works (GDMW) (1989) titled *'Value Engineering report: GDMW Standard Mosques: Various Locations'*, (iii) unpublished research by Fahad Al-Madhi (n/d) titled *'Design Criteria for*

Mosques' for the General Directorate of Military Works (GDMW), (iv) a book by Mokhtar and Ahmadi (2005) titled *'Design guidelines for ablution spaces in mosques and Islamic praying facilities'* published by the American University of Sharjah and, (v) a book by Kahera, Abdulmalik & Anz (2009) titled *'Design Criteria for Mosques and Islamic Centres Art, Architecture and Worship'* published by the Elsevier Architectural Press Imprint. In addition, the results of Ardalan's (1980) survey of historic Mosques was an invaluable base reference for understanding how the religious requirements of the Mosque were met by the early Muslims.

7.3.2.1 Introduction

This introduction provides a brief explanation of the religious requirements of mosque design, which any designer, whether Muslim or non-Muslim, must understand to design a Mosque.

7.3.2.1.1 Types of prayers

Every mature Muslim, whether male or female, is required to pray five times a day. These daily prayers can be conducted individually or in a group. When possible, Muslim males are required to pray in groups in a mosque. Women, on the other hand, can and often do, perform their prayers in their homes. The Arabic name of each of the five daily prayers is related to a time of day, because each prayer is scheduled at a specific time throughout the day and night. The five prayers are (in order), dawn (*Fajr*), noon (*Dhuhr*), mid-afternoon (*Asr*), sunset (*Maghrib*) and nightfall (*Ishaa*).

Friday Prayer: every Friday, a special prayer is performed, which replaces the *Dhuhr* (or noon) prayer. On Fridays, supplicants begin to converge on the Mosque by mid-morning, well ahead of the actual scheduled time for the prayer. The first of two calls to prayer (*Athan*) is up to an hour before the prayer and the second begins when the Imam steps up to the *Minbar*. The Imam gives a *khutbah* (lecture or speech) once the second *athan* is completed. The Imam's *khutbah* can vary in length, with the lecture or sermon given by some Imams taking much longer than others. In general however, a *khutbah* rarely exceeds an hour. Following the *khutbah,* the

actual Friday prayer begins. Being a special day of prayer (much like Sunday in Christianity), the congregation at any given mosque on a Friday at noon is larger than at any other time, on any other day of the week. In addition, not all mosques host Friday prayers (the ones that do are commonly called Friday Mosques) and so worshippers will congregate at the closest 'Friday Mosque', thus increasing the number of worshippers even further.

Eid Prayers: the two religious holidays of *Eid Al-Adh'ha* following the *Hajj* (pilgrimage to Makkah), and *Eid Al-Fitr* after the holy month of *Ramadhan* (the month of fasting) have special prayers. For these special *Eid* prayers, all Muslims in the area (men, women and children) wanting to pray, (it is not compulsory for all Muslims, as long as some do adhere to it), congregate at a nearby open space after dawn. After sunrise, the *Eid* prayers commence. Once the prayers have been completed, the Imam gives a special *Eid Khutbah*. In instances where it is not possible to have a large enough gathering to accommodate the entire population of the area (in cities, for instance) several sites are used. It is not uncommon, when Friday Mosques are large enough, to gather for *Eid* prayers there.

Ramadhan Prayers: during the holy month of *Ramadhan*, in addition to the five daily prayers, Muslims perform *Taraweeh* prayers at night after the *Isha* prayers. While the five daily prayers are compulsory, *Taraweeh* prayers are not. Many Muslims do, however, choose to attend these special prayers and it is usual for a special place to be made available for women. The *taraweeh* prayers are usually relatively long and can last for between an hour and two hours. The last ten nights of Ramadan are marked by *Qeyam* prayers in addition to the regular five daily prayers and the *taraweeh* prayers. Like the *taraweeh* prayers, the *qeyam* prayers are not compulsory. The *qeyam* prayers are performed after midnight and can last between two and three hours.

7.3.2.1.2 Types of Mosques

Mosques have been categorised in various ways, according to their size (Ibrahim, 1979; Kahera et al., 2009; Value Engineering Section, 1989), their typology (Ardalan, 1980; Frishman et al., 1994), and their usage (Al-Madhi, n/d; Frishman et al., 1994; Ibrahim, 1979; Value Engineering Section, 1989) to name just a few categorisations. One useful and reasonable way is to categorize mosques by incorporating both size and usage. By so doing, we can categorise mosques into 'daily' and 'Friday' residential mosques, commercial mosques, airport mosques and so on, because the size and usage vary and differ from one to another. This method of categorisation can aid the designer, because the usage of a residential mosque, for example, has a peak at *Fajr*, while in *Dhuhr* it reaches the minimum. In contrast, a mosque in a working area has a peak at *Dhuhr* and minimum in *Fajr*. Mosque types and usage patterns can be broadly broken down in the following way.

Residential mosques: as the name suggests, these are located in residential areas. Usually the washing areas can be designed to be relatively small, because most people prefer to perform their ablutions at home. The need for a space for women is not great, because it is only during the holy month of *Ramadan* that women are likely to perform their prayers in a residential mosque. The space requirements (Mosque capacity) are directly related to the number of houses that the mosque is expected to serve. The number of people flowing through a residential mosque is at its lowest at *Dhuhr*. The category of 'residential mosques' can be further sub-divided to give the designer even more valuable information regarding the usage and size of the mosque.

Residential daily mosques: the categorisation of a mosque as a daily mosque reflects that that particular mosque's location is at the centre of a residential area or group of houses with a population of between 500 and 1500. The daily mosque should not usually be located any further than a walking distance of about 150 to 200 metres from the residential clusters (Ibrahim, 1979).

Residential Friday Mosques: in areas where the population ranges between 3000 and 8000 inhabitants, Friday mosques are built in the centre of the neighbourhood. The Friday mosque is usually the dominant feature of such areas. It is recommended that a Friday mosque be no further than 250 to 200 metres away from residential areas (Ibrahim, 1979).

Commercial mosques: mosques designed to serve commercial areas have relatively large washing areas. In addition, a women's praying area and a women's washing area are required. The flow of people at a commercial mosque is at its lowest during the *Fajr* prayers. Commercial mosques, like residential mosques, can also be sub-divided to give the designer a greater insight into the requirements of the mosque being designed;

- Commercial daily mosques
- Commercial Friday mosques

Private mosques: built specially to serve specific groups of people, private mosques can be located in any place. For instance, large office complexes, military bases and universities will all have private mosques. The prayer hall area is directly related to the number of Muslim occupants who are expected to be present during any of the prayer times. These calculations for capacity take into consideration the availability of any other mosque in the area.

7.3.2.2 Mosque architectural design criteria

7.3.2.2.1 Form, shape and size

Mosques can take any form or shape provided they meet the religious requirements of offering a pleasant shelter for people to pray in. In addition, the most economical shape and structure are sought with regard to life-cycle costs.

The component and generic forms that constitute a mosque have maintained the functionality and the meaning throughout the history of the development of the Mosque, although their architectural forms have

undergone many developments that have been influenced by the ecology and the culture of the region where they exist (Frishman et al., 1994). Yet, there exist many common misconceptions about the shape and form of mosques. Many people believe that a dome and pillars, reflecting a common notion of 'Islamic Architecture', are a necessary feature of a mosque. It is not difficult to trace this misconception to the historic development of mosques. When the earliest mosques were built, the most convenient material to use was bricks, and the easiest way to build a roof using bricks is by erecting a dome shape. A dome constructed using bricks can only have a limited diameter, however, and in addition, every dome requires a number of columns to support it. As a result, mosques were built as a series of domes and columns (Al-Madhi, n/d).

The development and availability of steel and reinforced concrete structures in buildings offered the opportunity for designers to incorporate longer spans for roofs and floors into their designs. Theatres and gymnasiums (areas where large numbers of people congregate) were built using these new materials. Surprisingly, mosque design did not incorporate these developments in the construction materials. The irony, as Al-Madhi points out, is that some mosques have domes built from reinforced concrete. This, of course, is much more expensive and more resource intensive than building flat roofs (Al-Madhi, n/d).

The size of the praying-age population that a mosque is designed to serve dictates the size of a mosque. In most Muslim countries, areas with a population of 1000 Muslims, are on average found to be made up of 20% children who are below the age of praying and 40% females who are not religiously bound to pray in a mosque (although it is optional to pray in a mosque and some women do so). The remaining 40% are males who pray in mosques (Ibrahim, 1979). Not all 40% of the male population will always be present at the mosque, therefore, a space allocated for 40% of the population is usually considered to be well able to cope with any women and children who chose to attend the prayers in the mosque.

7.3.2.2.2 Zoning of the Mosque:

As mentioned earlier, mosques usually have two main components, the prayer hall and the courtyard. It is, however, helpful to divide a mosque into three zones, the prayer hall, the courtyard and the 'auxiliaries'. This aids the designer further in gaining a clear picture of the various required functions of a mosque.

7.3.2.2.2.1 Prayer hall

Usually, the prayer hall is a rectangular-shaped room. Its axis is usually aligned to the *Qiblah* (the direction to the *Kaa'ba* in Makkah). The *Mihrab* (the niche in the wall indicating the *Qiblah*) and in the case of a Friday mosque, the *Minbar* (the pulpit from which the Imam speaks), is usually located at the centre of the front of the room. One or more entrances lead into the prayer area and, in Friday mosques, the Imam has a special entrance at the front of the hall. Female members of the congregation during the month of *Ramadhan*, will have a small space at the back of the main prayer hall partitioned off for them in daily residential mosques. The female area in larger mosques can, however, sometimes occupy a specially allocated hall, usually on a mezzanine level, with a separate entrance and ablution area.

Because each prayer needs a certain amount of space for praying, the dimensions of the prayer hall can be directly calculated from the number and length of rows of prayers expected at the busiest times. There is a need, however, for a clearly-defined criterion to follow, because the combinations of rows and lengths of rows can produce a vast number of different dimensions. For instance, a prayer hall designed to accommodate a hundred prayers could be designed so that there is one row of one hundred prayers, two rows of fifty prayers, four rows of twenty-five prayers and so on. The other constraints faced by the designer (site, economic cost and so on) will naturally influence the criteria chosen and the extremes, for instance, one row of one hundred prayers will not occur. As prayers generally feel there is more value in praying closer to the front of the mosque, the general preference is to have the longest possible rows in the

front of the mosque to accommodate this. All consecutive rows should be either of equal length to the first row or can, in some cases, be shorter. Thus, rectangular shaped prayer halls are found to be the most appropriate, because rows of equal lengths can be achieved. The ideal situation is to have the longer side of the rectangle of the prayer hall facing the Qiblah direction towards the Kaa'ba, so that the longest possible rows of prayers can be accommodated.

Al-Madhi, Ibrahim and Kahera (Al-Madhi, n/d; Ibrahim, 1979; Kahera et al., 2009; Value Engineering Section, 1989) found through observation, practice and research that a suitable distance between rows is about 1.20–1.60 m. This distance allows prayers enough space to prayer comfortably without wasting valuable space. Trying to fit rows closer together results in the comfort of the worshippers being lost and providing more than 1.20–1.60 m for each row adds unnecessarily to the total cost. It is recommended by Al-Madhi, Ibrahim and Kahera to add 0.1 m to the front of the first row to allow for space for the first row to prostrate comfortably. It is also recommended that 0.5 m be added to the last row to provide circulation behind the last row. Al-Madhi, Ibrahim and Kahera also found that the optimal width of the space required for praying is between 0.60–0.80 m. Thus, the area required for each prayer is in the region of 0.72–1.2 m^2. Commonly 1 m^2 per person is used as a rule of thumb when calculating the required areas of the prayer hall. Thus, based on the projected maximum capacity at the busiest times, the minimum area of space required for the prayer hall can be calculated (Al-Madhi, n/d; Ibrahim, 1979; Kahera et al., 2009; Value Engineering Section, 1989).

It is worth noting that all the calculations discussed above assume that the prayer area is a clear space. If columns or other obstacles exist, the design will need to be altered. For instance, it is unacceptable for a column to cut a row up and so the row would in all probability have to be moved either in front or behind the column and this would add to the space required for the prayer hall.

7.3.2.2.2.2 Courtyard

The courtyard is used mainly for social and environmental reasons, but it also provides a space to accommodate any unexpected increases in the number of prayers who congregate at the mosque. The portico, which is located on the *Qiblah* side of the courtyard, is used as an outdoor prayer area during pleasant weather. The courtyard also plays an architectural organising component linking all the Mosque components by placing them around it.

In Ardalan's (1980) survey, the courtyard appears in 93% of the mosques that were analysed, reflecting the importance of the function of integration between all of the mosque components that the courtyard provides. Aazam (2007) conducted a socio-spatial organization analysis of 12 selected mosques that represent Ardalan's six Muslim ecological regions. Aazam finds that the courtyard is the most integrated space, which again reflects its importance, not only as the hub of the mosque system, but also as the main space for the mosque's function of providing a place for social activity.

Most of the studies on mosques that intended to provide design criteria for the mosque courtyard suggest a ratio of between 20–60% of the area allocated to the prayer hall. It has been noticed by the author that the studies and regulations for the planning and design criteria of a mosque's courtyard, which are even adopted by government authorities, are not actually based on any reasoned or logical grounds. The somewhat random and flexible guidelines for the courtyard's design may result from the fact that the courtyard, as an open space, can, if the site is large enough, be used for various activities when the weather permits and more importantly, can also be used to accommodate additional prayers when the need arises. From this point of view, the courtyard can be considered as having a secondary social and an environmental function that is governed by the available area of the whole site.

Taking the environmental significance of the courtyard in mosque architecture into consideration, the courtyard is designed to provide,

among other functions, a space for social activities, and a source of natural lighting and fresh air for the prayer hall. Unsurprisingly, the climate of a region greatly influences the size of the courtyard. The courtyard of mosques found in Arabia, Persia (Iran) and India are spacious, while those found in Turkey and Central Asia are smaller, and in some extremely cold regions, the courtyard has completely disappeared. The porticos are designed to provide shaded areas around the edges of the courtyard of most of the mosques built in Egypt and the eastern Muslim countries. In the western Muslim countries and Andalusia, the porticos are covered for protection against heavy rain.

7.3.2.2.2.3 Ancillary spaces

Any additional annexes, which can vary widely from one mosque to another, come under this zoning category. Some of these annexes are essential to the mosque, such as the ablution area and the Minaret. Other annexes are optional depending on the size, function and location of the mosque. Examples of additional annexes include function halls, libraries, classrooms, caretaker's suites, storage spaces, a residential area for the Imam and *Mu'adhin* (caller to prayer), shops, gardens and fountains and parking spaces and so on.

7.3.2.2.2.3.1 Ablutions area

This area within a mosque is very important, because every prayer has to perform wudhu (ablution) to cleanse himself (or herself) before every prayer. The performance of wudhu, involves the washing of hands, face, arms (up to the elbows), head and feet. The ablutions area can get very crowded just prior to the prayer, especially if it has not been properly designed. There is no direct proportional measurement that helps to determine the correct size of the washing area, because the percentage of people performing *wudhu* in the mosques (as opposed to at home before coming to the mosque) cannot be precisely determined. As mentioned earlier, in a residential area, most people perform *wudhu* in their homes, while in commercial areas they use the mosque's washing area.

It is strange that the ablution area, required in mosques, is not governed by clear guidelines. Nevertheless, the focus of most guidelines for mosque design, made available in many Islamic countries, centres around the design of the prayer areas. Toilet design guidelines for mosques follow the standard guidelines for toilets and are readily available to most designers. As Mokhtar and Ahmadi point out, the lack of design guidelines for the ablutions area means that designing these areas so that they are safe and comfortable for the prayers is very challenging (Mokhtar & Ahmadi, 2005). There are a few basic rules-of-thumb that can be used, however. Ibrahim for instance, recommends the use of a 1:25 ratio for residential area mosques (Ibrahim, 1979). The ratio is relatively small in residential area mosques because, as mentioned before, most people perform their ablutions at home prior to coming to the mosque for prayer.

Mokhtar and Ahmadi (2005), on the other hand, cite a paper by Nofel (1999) titled *'Design criteria for mosque architecture'* (Proceedings of Research on Mosque Architecture, College of Architecture and Planning, King Saud University, Saudi Arabia, 5, pp. 75–94) (in Arabic), which describes a pilot study for residential area mosques in Egypt. The study conducted a survey of ablution area use. The results could be seen in curves rather than fixed ratios. As can be expected, residential areas and commercial areas were found to have different requirements. The minimum ablution area requirements for a residential area mosque with a capacity of 225 users was found to be two units; however, Nofel recommends an ideal ratio of not less than 1:65, with the ratio increasing to 1:40 for a mosque with a capacity of 600 users. Once the capacity of a mosque increases further, the ratio of ablution units to users decreases again to about 1:55. For mosques in commercial areas, the minimum requirement for ablution areas is larger, because fewer people are able to perform their ablutions before coming to the mosque. Nofel's study recommends no less than three ablution units for a mosque with a capacity of up to 200 users. The ideal ratio recommended starts at 1:40 and increases to 1:32 for a mosque with a capacity of 800 users. Once again, the

ratio of ablution areas to prayers decreases to 1:45, once the mosque capacity reaches 2000 users.

Mokhtar and Ahmadi (2005) monitored the prayer areas in several shopping malls and observed the flow of prayers coming to pray within a time span of five minutes before the start of the prayer until the end of the prayer. They note that the percentage of those who needed to perform ablutions before the prayer was about 60% of the prayers arriving every minute. The number of prayers arriving at the prayer area varies every minute and was represented with a curve. The results of Mokhtar and Ahmadi's observations were compared with a variety of prayer areas each with different capacities and calculated, based on a proposed equation resulting in a ratio between 1:11 to 1:15. It is not clear if the study covered each of the five prayer times, but we can assume that, at least, the peak times were selected. The type of society where the study took place is not mentioned and this would also affect the results, because the percentage of people who actually pray every one of the five daily prayers varies from one nation to another and from one community to another.

The only design constraint of the toilet area design comes from religious guidance that states that Muslims should not either face towards or turn their backs to the Qiblah direction while in the toilet. Therefore, the WC's should be placed perpendicular to the Qiblah direction.

7.3.2.2.2.3.2 The Minaret

Adjacent to the Mosque, the minaret serves as a place from which the Mu'adhin calls worshippers to prayer. The minaret is a tall narrow structure that allows for the voice of the Mu'adhin to be carried over great distances. In the early days of Islam, minarets were not a feature of mosques. The call to prayer was simply made by the Mu'adhin from the roof of the mosque or from any tall building in the vicinity. As with the dome, however, the minaret has developed into a symbolic representation of a mosque, and now has the dual function of carrying the call to prayer over greater distances and as a landmark for the identification of a mosque.

7.3.2.3 Mosque acoustics design criteria

7.3.2.3.1 Defining the problem

The acoustics of a mosque are of vital importance, because the voice of the Imam has to be heard by all prayers within the mosque and at busy times, and should also be heard by those following the prayers from outside the mosque. The main source of sound within a mosque is the voice of the Imam, whether in daily, Friday or Eid prayers. During prayers, the Imam faces the Qibla (with his back to the prayers) and for his sermon during Friday prayers, he faces the congregation — irrespective of which way he is facing, the congregation must be able to hear the Imam clearly. Imams are chosen on their capability of reciting the Holy Quran, and not on their ability to throw their voices; some Imams experience difficulties in delivering clear speeches and sermons that are both understood and heard by all those congregating. These difficulties can be compounded if the mosque is large and does not have well-designed acoustics. During the Eid Prayers, when the majority of the community gathers to perform the prayers together in an open space, the role of acoustics becomes even more vital.

Another source of sound emanating from a mosque is the call to prayer (Athan), performed by the Mu'adhin. Traditionally, the Mu'adhin would have climbed to the top of the mosque roof (or the roof of a tall adjoining building), or later (when they began to appear in mosque architecture) to the top of the minaret, so that the sound of the call to prayer would carry over greater distances and be heard by the entire neighbourhood. With the advent of technological advances, the acoustics associated with the call to prayer (that is, outside the mosque) have become greatly simplified, because amplifiers are almost always placed on top of the minarets. Not only does this mean that the call to prayer is heard over far greater distances, but it also means the Mu'adhin can actually perform the Athan from inside the mosque itself.

7.3.2.3.2 Design criteria

Clearly, when designing a mosque, the acoustics play an important role, because the voice of the Imam needs to be heard by the whole congregation gathered inside the mosque. In an ideal situation, the Imam should also be heard by those gathered outside following the Imam's prayers and sermons during busy, crowded times. As the splitting of the prayer areas by floor levels is not usually recommended (or preferred), except in the case of a female prayer hall, the design reverberation time and acoustic criteria for a mosque can be based on rooms of similar sizes designed to host gatherings of people assembled primarily to listen to a speaker. The design reverberation time in a mosque can thus be assumed to be within the same range as that of a lecture or conference room, which is in the range of 0.9 to 1.1 seconds. The acoustic criteria of a mosque can similarly be assumed to be in the range of that of a small conference room or a classroom. The Preferred Noise Criteria (PNC) for a mosque is thus between 30 to 35 (Watson, Crosbie & Callender, 1997).

Although the need for the call to prayer (Athan) to be loud enough to be heard by all the neighbouring houses has been greatly simplified by technology, there is an added requirement to make sure it is not so loud that it will conflict with the Athans from other mosques in the area.

It is worth noting that all other activities within the mosque (for instance shops, classes and so on) will cease during prayer time, and thus, so will their associated noises, making it easier for the Imam to be heard. In general however, the approach taken with regard to noise control in a mosque is very similar to that taken in other buildings.

7.3.2.4 Mosque lighting design criteria

7.3.2.4.1 Defining the problem

Although the primary function of a mosque is to provide a pleasantly sheltered area in which Muslims can pray, the members of the congregation also often sit and read the Holy Quran in the mosque. Thus, not only is sufficient lighting in a mosque necessary for circulation and seeing the

Imam as he leads a prayer and delivers his sermon (for Friday prayers), but it also needs to be sufficient for reading. Generally, members of the congregation sit on the floor of the prayer hall and read the Holy Quran before prayers. The height of the task that requires light (reading) can thus be taken to be slightly above floor level (between 0.1–0.3m).

7.3.2.4.2 Design criteria

There are no hard and fast rules or guidelines for the illumination of the prayer hall in a mosque. Generally, the lighting system of a mosque should be designed to allow for easy unhindered circulation by the congregation, easy visual contact with the Imam at the front of the prayer hall and should be sufficiently bright to allow for unhindered reading of the Holy Quran.

The colour of the light used is not governed by any regulations, but the level of illumination required is usually assumed to be on a par with that required for reading poor reproductions in a general office, which is in the region of 1500 lux (Watson et al., 1997).

The spacing of the sources of light is usually determined by the preference to illuminate each row of prayers independently. Illumination does not necessarily need to be at ceiling level and is thus sometimes lowered to provide the best light for reading or according to the most economical solution. When the mosque is not at full capacity, only the first rows require illumination and this, too, influences the design of the lighting system in the prayer hall of a mosque. Lighting in other areas of the mosque (such as the toilets, library) is governed by the prevailing standard practice when illuminating the same sorts of areas in any building.

Mosques can host large gatherings of people, thus, lighting systems that produce excessive amounts of heat are avoided. This is a particularly important consideration in the hotter climates. From the perspective of providing the congregation with a safe environment, sufficient lighting should be provided for circulation and safety lighting with sufficient standby capacity, and should be provided at door exits.

7.3.2.4.2.1 Illumination during daylight hours

The illumination of a mosque during daylight hours is as important as illumination during the night, because two of the five prayers occur when there is daylight. The use of natural (sun) light as a source of illumination during the day can provide substantial energy savings, while also adding to the heat of the building. Clearly, the climate and environment in which a mosque is built greatly influence its daylight design. Although, in colder climates the heat from sunlight is clearly a benefit both economically and environmentally, in hotter, sunny climates (the climatic regions of most Muslim countries) the additional heat produced by natural light leads to a need for more energy to cool the building. The energy used to cool the building can potentially offset or even negate the economical and environmental gains made by using natural light.

Because the time taken to perform the two daylight prayers is just a small fraction of the whole day, the energy used to artificially illuminate the prayer hall during prayer times will, in all probability, be a fraction of the energy that would be required to cool the same area. In addition, from the perspective of the worshippers, direct sun is (especially in hot climates) not very comfortable. A final consideration in designing a mosque for daylight, which is equally relevant to both hot and cold regions, is that worshippers are encouraged to concentrate on their prayers and large windows designed to let natural light into the prayer hall may also provide views that could distract the congregation.

7.3.2.4.2.2 Exterior lighting

A mosque is illuminated on the exterior at its entrance and its minaret. The entrance has to be sufficiently well lit to allow easy and safe access and exit from the mosque. The walkways leading from the mosque gates to the entrance into the prayer hall should also be adequately lit for young and old (possibly with bad eyesight) to be able to see without trouble where they walk.

The minaret of a mosque is usually illuminated to guide worshippers (especially those not familiar with the area) towards the mosque for

prayers. Frequently, floodlighting is used and is appropriate for the purpose. The need for illuminating the minaret as a guide towards the mosque is more important for *Isha* (nightfall) than *Fajr* (dawn) prayers, because at dawn, most people are in their own areas and know where their mosque is located.

7.3.2.5 Mosque thermal control design criteria

7.3.2.5.1 Defining the problem

Mosques are used for short periods five times a day, therefore, conventional thermal control criteria cannot be usefully applied in their design. Because most Muslim countries have hot climates, pre-cooling is a prime example of the conventional methods of thermal control being unsuitable for mosque design. Muslims are encouraged to go to the mosque before prescribed prayers to perform voluntary prayers or to read the Holy Quran. If a mosque is pre-cooled before a prescribed prayer, those worshippers who have arrived early will be subjected to the discomfort of the pre-cooling. If pre-cooling is not carried out, however, the heat build-up in the prayer hall during peak times will also cause discomfort for the worshippers. The design of the thermal controls of a mosque seeks to find a balance between the extremes of the uncomfortable cold of pre-cooling and the uncomfortable heat of a prayer hall filled to capacity.

7.3.2.5.2 Design criteria

Clearly, the design of the thermal control for a mosque (much like any other building) must take the local climatic conditions into consideration. As a general rule, the projected duration of use at each prayer time is about one hour, beginning when the call to prayer (*Athan*) is made up until the last person leaves the mosque after the prayer. As mentioned earlier, the lighting design of a mosque will influence the heating or cooling requirements. Ideally, a mosque should have an optimum balance of window area for natural lighting with no direct sunlight and energy gain/loss and offer as little distraction to the congregation as possible. No other electrical appliances or machines that might significantly affect the thermal levels of the mosque are normally used.

Most mosques, as mentioned earlier, are usually rectangular and there are no restrictions on where the HVAC equipment is placed, provided it does not interfere with, or hinder, any of the activities taking place in the mosque. It is usually recommended that the electric and water services for the mosque should be integrated in the same area of the mosque where the heating and air conditioning ducts, piping and equipment have been located.

Although the courtyard and ancillary spaces within a mosque are sometimes used for other activities, in keeping with religious principles, the main mosque space is not converted for other uses. Most of the spaces within the mosque and specifically the prayer hall (which takes up the most space) are thus very stable and predictable throughout the year in terms of use, expected capacity and so on. The fact that mosques (and again more specifically the prayer hall) are very sparsely furnished also affects the design of thermal controls for a mosque. Apart from carpeting, the thermal controls of the space will thus only be greatly affected by the circulation and gathering of the congregation and, of course, the climate and weather.

Another factor to consider in the design of thermal controls is that mosques very often have one or more swinging doors; one main door leading into the prayer hall for the congregation to use and another at the front of the prayer hall used by the Imam. These doors are often left wide open, adding to the problem of thermal control. While the loss or gain of heat is impacted by the open doors, the design of ventilation for peak times when the prayer hall is at full capacity is critical. The flow patterns of the congregation are clearly important considerations when studying the heating and cooling loads. On a Friday, for example, the cooling or heating load will be at its height. The size and location of the mosque usually dictates the type of heating and air-conditioning system that is used. For instance, in a small mosque in hot climates, window air conditioning units can be used, but these units would be inappropriate for a larger mosque. In hot climates, automatic controls are recommended during the summer, especially when the mosque does not host other activities outside of the scheduled prayers.

There are also no regulations or restrictions governing the materials used in building a mosque; thus, the most economical option, based on a life-cycle basis and taking account of thermal control considerations, should be chosen.

7.3.3 Illustrative project's design context

The project chosen to illustrate the HEAD system architecture is based on the General Directorate of Military Works of the Saudi Arabian Armed Forces (GDMW) requirements for a standard Friday Mosque. The requirement is for a Friday mosque to serve a community with a population of 7500. The mosque is required to have a capacity of 3000 prayers in total prayers, a main prayer hall capable of hosting 2500 prayers and a female prayer hall with a capacity for 500 female prayers. The assumption is that the mosque is located in a local commercial centre of a residential neighbourhood. Thus, the mosque is intended for a mixture of both residential and commercial use. In the interests of maintaining simplicity and to focus on the design system proposed in this research, the site is assumed to be flat and open, with no structures adjoining the site. The site is also assumed to be substantially larger than the building but, for the considerations of the building orientation design constraints in relation to the site orientation and boundaries, it is suggested using a general standard city block (Robbins & El-Khoury, 2004) and follow the dominant city grid of the selected cities, Riyadh, Saudi Arabia (latitude 24° 40' N, longitude 46° 43' E) and Brisbane, Australia (latitude 27° 28' S, longitude, 153° 1' 0" E) vs. Makkah, Saudi Arabia (latitude 21° 25' N, longitude 39° 49' E) Sours; Google Earth.

7.3.3.1 Design constraints

Following Lawson's (1997, 2004, 2006) categorisations of design constraints into external constraints, internal constraints and self-imposed constraints, the context of the selected GDMW standard mosque design was analysed and the initial design constraints were identified and classified. The main constraints that govern the Mosque design in general are highlighted in the following paragraphs.

7.3.3.1.1 External design constraints

7.3.3.1.1.1 Environment

- Site
- Location
- Dimensions
- Area
- Orientation
- Access
- Topography
- Nature
- Neighbouring
- Views

7.3.3.1.1.2 Local weather:

Local weather data is readily available from many resources such as Energy Plus.

7.3.3.1.1.3 Laws and Codes

7.3.3.1.1.3.1 Building regulations

Local authority laws and regulations for the planning of religious and community facilities are considered and followed. The local regulations do not generally pose any major design constraints. The only exceptions are the local laws and regulations that relate to the functionality and performance of the building.

7.3.3.1.1.3.2 Building codes

Local building codes for religious and community facilities are considered and followed. Similar to the local authority laws and regulations governing building regulations, the local building codes are not generally seen as a

major design constraint, except fore the building codes that relate to building functionality and performance.

7.3.3.1.1.3.3 Design standards

General building design standards are mainly considered to govern the spatial requirements for their direct relations to the building's function and performance.

7.3.3.1.2 Internal design constraints

7.3.3.1.2.1 Spatial requirements

7.3.3.1.2.1.1 Room list

Table 7-1 Mosque Room List (Section 7.6) is developed in accordance with the GDMW requirements for a standard mosque with a capacity of 2000 prayers. Some enhancements are added to the room requirement, based on the research of historic mosque architecture and the general mosques design criteria briefly discussed earlier in the section on the background of mosque architecture.

The external spaces surrounding the Mosque are included in the room list for a later purpose, when setting up the adjacency requirements between the spaces. Some of the rooms have certain requirements that call for them to be allocated on a particular side of the building, such as the ablutions and toilet areas that are preferred at the back of the mosque.

7.3.3.1.2.2 Spatial relations (adjacencies)

The adjacency data (represented in a graph) can be processed as either true or false, which means that it is impossible to define an adjacency priority. However, the adjacency rules and requirements can be set by the designer, who would base these on the functional relations between the spaces of the building. Different graphs generated from weighted adjacency matrices can be used to provide the designer with alternative solutions for the adjacency requirements. Based on Tarjan's conception of depth-first research (Wu et al., 2004), all possible paths between the nodes of a weighted graph, according to a weighted adjacency matrix, are searched for.

7.3.3.1.2.3 Design criteria

Form and shape, Lighting, Acoustics and Thermal design criteria were discussed earlier**Error! Reference source not found.**.

7.3.3.1.2.4 Mode of fabrication

See section (7.7) for GDMW Specifications.

7.3.4 Illustrative project's design concept

Based on Lawson's works (Lawson, 1997, 2004, 2006) the design concept and the design schema are influenced by the designer's preconceptions and are thus considered to be self-imposed constraints added to the design problem.

The design concept for the Mosque design project is inspired by a combination of selected elements from Rowe (1998), Kolarevic (2003) and Flanagan's (2005) approaches. The concept adapts Rowe's lateral 'analogies', 'environmental relations', 'typology' and 'formal language', with Kolarevic's 'parametric design', 'genetics' and 'performative morphogenesis', in addition to Flanagan's 'generative geometry' and 'generative manufacture'.

Through observation and with reference to Rowe's lateral analogies, typology and formal language (1998), we note that all the generic forms are included within the two main parts of the mosque, the prayer hall and the courtyard. Consequently, we can distinguish mosques that have only a prayer hall, hosting the generic forms associated with it and mosques that combine the prayer hall with a courtyard, which include the generic forms associated with both.

Aksamija (2007) cites Rafael Moneo as stating that, 'the work of Architecture is irreducible to any classification', suggesting that typology is based on the continuous development of shared characteristics (Aksamija, 2007), which always results in a fitter generation of types. Examining mosques from a genetic development perspective and based on Moneo's concept of developing types, we note that mosque typology has, indeed,

developed from a gene pool of generic forms that follows a typology specific to the ecological regions. For instance, the first mosques were built as simple hypostyle prayer halls. The Mihrab, the pulpit '(Minbar)', the minaret and the courtyard were introduced later, followed by other generic forms that were influenced by the various ecological Muslim regions. The emphasis of a certain generic form within the combination of generic forms, with respect to the principles of spatial organisation, came about in answer to the specific needs and architectural language of a particular region. Mosque architectural development can thus be regarded from a genetic perspective where certain generic forms were introduced by an ecological region, such as chromosomes, resulting in a distinguishable typology. For example the dome, the Iwans and the pyramidal roof were all introduced as genotypes, which resulted in a phenotype typology. Regional genes or generic forms are adopted by other regions, fusing into another generation of typologies.

Examining the mosque from the perspective of the developing typologies, we can identify six regions with six dominant typologies. These typologies are all influenced by their environment and culture, resulting in a set of guidelines or a set of shared characteristics specific to each typology. The characteristics of each typology can be formalised by different schemas, where each schema produces several successful permutations.

7.3.5 Illustrative project's design schema

When designing functional, technologically-sound and economical solutions, a designer does not have to give up aspiring to produce a solution that expresses his or her creativity. Process-oriented design, see Sections (3.2.3) and (4.6), places a greater emphasis and value on the aesthetics of the process and the organisation of the designed product. Process-oriented design is a relatively new school of technological thought that, while based on mathematics and algorithms, also adopts a new definition of beauty. Process-oriented design places a high value of aesthetics on the design process, which it regards as being based on the dynamic processes of the

constructive patterns, rather than on the object orientation of more traditional design processes.

The 'design schema', discussed in Section (4.6), is taken by the author and this research to be the individual design model or framework used by each designer in an attempt to solve the design problem being faced. The designer's guiding principles, primary generators and personal schema (collectively the designer's pre-conceptions) influence the way in which the designer perceives the constraints of each specific project. They can, in turn, often be affected and changed by each new problem that the designer faces. The design schema is thus unique, not only to every designer, but also for every design problem faced by every designer. The design schema (in addition to the 'designer's schema) is dynamic and changeable and this quality means that the designer can inject his or her creativity into their design solution.

It is intended that at the heart of the proposed design system, the genetic evolutionary algorithm will be the main engine generating design solutions. Models that have been successfully evolved will be the optimisation using genetic evolutionary algorithms that will simulate environments interpreted from the external and internal constraints of the design problem.

7.4 Appendix D

Mosque DCG (pseudo code)

Pseudo Code of Definite Clause Grammar (DCG) descriptions for mosques, based on historic typologies:

Mosque → prayer_hall, courtyard; prayer_hall

Mosque → prayer_hall, courtyard
prayer_hall → worship_area, gateway, appendixes
worship_area → roof, walls, floors
roof → hypostyle, hypostyle_dome_accent, hypostyle_domical_vaulting; central
hypostyle → [hypostyle]
hypostyle_dome_accent → [hypostyle_dome_accent]
hypostyle_domical_vaulting → [hypostyle_domical_vaulting]
central → dome; pyramid
dome → ottoman_dome; indian_dome
ottoman_dome → [ottoman_dome]
indian_dome → [indian_dome]
pyramid → [pyramid]
walls →Qiblah, right, left, back
Qiblah → Mihrab, Minbar, imam_entrance, fenestration
Mihrab → [Mihrab]
Minbar → [Minbar]
imam_entrance → [imam_entrance]
fenestration → windows
windows → [windows]
right → entrances, fenestration
entrances → doors
doors → [doors]
fenestration → windows
windows → [windows]
left → entrances, fenestration
entrances →doors
doors → [doors]
fenestration → windows
windows → [windows]
back → entrances, fenestration
entrances → doors
doors → [doors]
fenestration → windows
windows → [windows]
floors → [floors]
gateway → plinth, entrances

plinth → steps, platform
steps → [steps]
platform → [platform]
entrances → doors
doors → [doors]
appendixes → storage,;
storage → Quran-storage, carpet_storage
Quran-storage → bookshelves
bookshelves → [bookshelves]
carpet_storage → racks
racks → [racks]
courtyard → court, appendixes
court → Iwans,; portico, minaret, gateway
Iwans → [Iwans]
portico → columns, roof
columns → [columns]
roof → hypostyle, hypostyle_dome_accent, hypostyle_domical_vaulting; central
hypostyle → [hypostyle]
hypostyle_dome_accent → [hypostyle_dome_accent]
hypostyle_domical_vaulting → [hypostyle_domical_vaulting]
central → dome; pyramid
dome → ottoman_dome; indian_dome
ottoman_dome → [ottoman_dome]
indian_dome → [indian_dome]
pyramid → [pyramid]
minaret → [minaret]
gateway → plinth, entrances
plinth → steps, platform
steps → [steps]
platform → [platform]
entrances →doors
doors → [doors]
appendixes →storage, madrasa, ablution_place, care_taker_suit, services
storage →janitorial_storage, carpet_storage
janitorial -storage →shelves
shelves → [shelves]
carpet_storage →racks
racks → [racks]
madrasa →imam_office; library; classroom
imam_office →office
office → [office]
library → [library]
classroom → [classroom]
ablution_place → male, female
male →ablution_area, toilets
ablution_area → (n) ablution_stand

ablution_stand →ablution_stand]
toilets →wc_cubicles, wash
wc_cubicles → (n) [wc_cubicles]
wash → (n) [sinks]
female →ablution_area, toilets
ablution_area → (n) ablution_stand
ablution_stand → ablution_stand]
toilets → wc_cubicles, wash
wc_cubicles → (n) [wc_cubicles]
wash → (n) [sinks]
care_taker_suit →bedroom, kitchentt, lavatory, sitting
bedroom →bedroom]
kitchenett → kitchenett]
lavatory → wc, shower, sink
wc → wc]
shower → shower]
sink → sink]
sitting → sitting]
services → mechanical_room, electrical_room
mechanical_room → mechanical_room]
electrical_room → [electrical_room]

Mosque → prayer_hall
prayer_hall → worship_area, minaret, portico, gateway, appendixes
worship_area → roof, walls, floor
roof →hypostyle, hypostyle_dome_accent, hypostyle_domical_vaulting; central
hypostyle → [hypostyle]
hypostyle_dome_accent → [hypostyle_dome_accent]
hypostyle_domical_vaulting → [hypostyle_domical_vaulting]
central →dome; pyramid
dome →ottoman_dome; indian_dome
ottoman_dome → [ottoman_dome]
indian_dome → [indian_dome]
pyramid → [pyramid]
walls →Qiblah, right, left, back
Qiblah →Mihrab, Minbar, imam_entrance, fenestration
Mihrab → [Mihrab]
Minbar → [Minbar]
imam_entrance → [imam_entrance]
fenestration →windows
windows → [windows]
right →entrances, fenestration
entrances →doors
doors → [doors]
fenestration → windows
windows → [windows]

left → entrances, fenestration
entrances → doors
doors → [doors]
fenestration → windows
windows → [windows]
back → entrances, fenestration
entrances → doors
doors → [doors]
fenestration → windows
windows → [windows]
floor →floor]
minaret → [minaret]
portico → columns, roof
columns → [columns]
roof → [roof]
gateway → plinth, entrances
plinth → steps, platform
steps → [steps]
platform → [platform]
entrances → doors
doors → [doors]
appendixes → storage,; madrasa, ablution_place, care_taker_suit, services
storage → Quran-storage, carpet_storage
Quran-storage → bookshelves
bookshelves → [bookshelves]
carpet_storage → racks
racks → [racks]
madrasa → imam_office; library; classroom
imam_office → office
office → [office]
library → [library]
classroom → [classroom]
ablution_place → male, female
male → ablution_area, toilets
ablution_area → (n) ablution_stand
ablution_stand →ablution_stand]
toilets →wc_cubicles, wash
wc_cubicles → (n) [wc_cubicles]
wash → (n) [sinks]
care_taker_suit → bedroom, kitchentt, lavatory, sitting
bedroom → [bedroom]
kitchenett → [kitchenett]
lavatory → wc, shower, sink
wc → [wc]
shower → [shower]
sink → sink]
sitting → sitting]

services → mechanical_room, electrical_room
mechanical_room → mechanical_room]
electrical_room → [electrical_room]

7.5 Appendix E

Grid generation

A programme developed by Gavin Munro to demonstrate the author's concept of the use of the regulating lines as a consistency controller to generate overlapping shifting grid for the chosen project.

```
''' Created on 14/09/2010
@author: munrog '''

from sage.plot.point import point
from sage.plot.line import line

V = VectorSpace(RR, 2)

class Point2:
    __slots__ = ['x', 'y']

    def __init__(self, x, y):
        self.x = x
        self.y = y
        self.p = V([x, y])

    def distance(self, other):
        c = LineSegment2(self, other)
        if c:
            return sqrt(c.v[0] ** 2 + \
                        c.v[1] ** 2 )
        else: return 0.0

    def isOn(self, L):
        #find intsn Point self.x, self.y and LS(p,v)
        if L.p == self: #projn will be 0
            return L.p
        else: #gav #debug #just get rid of this? #err!!!  No __get_item__
            diffv = V([self[0], self[1]]) - V([L.p[0], L.p[1]])
            #theta = arccos( diffv.dot(L.v) / diffv.magnitude() *
Lv.magnitude() )
            #theta = 0 iff cos(theta) = 1 <=> ratio of magn**2 = 1
            d2 = diffv[0]**2 + diffv[1]**2
            s2 = L.v[0]**2 + L.v[1]**2
            ratio = diffv * L.v / (d2*s2) #arccos diffv/d*s gives theta
            #gav Square root used to calc magnitude is an expensive
operation
            if ratio == 1:
                return self
            else:
                return None

class LineSegment2:
    __slots__ = ['p', 'v']

    def __init__(self, p2, v2):
        self.p = p2
        self.v = v2

    def _u_in(self, u):
```

```
        return u >= 0.0 and u <= 1.0

    def ray(pq):
        #return a Sage ray ?
        pass #V([pq[0], pq[1]-pq[0])

    def ends(self):
        #return a tuple of V(RR, 2) instances
        return [V(self.p), V(self.p) + V(self.v) ]

    def draw(self, c):
        g = Graphics()
        g += line2d([self.p, moved(self.p, self.v)], thickness = 1,
color=c)

    def intersect(self, ls2):
        A = self
        B = ls2
        d = B.v[1] * A.v[0] - B.v[0] * A.v[1]
        if d == 0:
            return None

        dy = A.p[1] - B.p[1]
        dx = A.p[0] - B.p[0]
        ua = (B.v[0] * dy - B.v[1] * dx) / d
        if not A._u_in(ua):
            return None
        ub = (A.v[0] * dy - A.v[1] * dx) / d
        if not B._u_in(ub):
            return None

        return V([ A.p[0] + ua * A.v[0], A.p[1] + ua * A.v[1] ])

class Line2:
    #Represent as (p, v)
    __slots__ = ['p', 'v']

    def __init__(self, p2, v2):
        self.p = p2
        self.v = v2

    def _u_in(self, u):
        return True

    #gav ################################
    def intersectP(self, P):
        L = self
        #find intsn Point P.x, P.y and LS(p,v)
        if L.p == P: #projn will be 0
            return L.p
```

```
        else:
            diffv = P - L.p
            #theta = 0 iff cos(theta) = 1 <=> ratio of magn**2 = 1
            #theta = arccos(diffv.dot(L.v) / diffv.magnitude() *
Lv.magnitude())
            #gav Square root used to calc magnitude is an expensive
operation
            d2 = diffv[0]**2 + diffv[1]**2
            s2 = L.v[0]**2 + L.v[1]**2
            ratio = diffv * (L.v) / (d2*s2) #arccos diffv/d*s gives theta
            if ratio == 1:
                return P
            else:
                return None

    def isnLS(self, Seg):
        L = self
        d = Seg.v[1] * L.v[0] - Seg.v[0] * L.v[1]
        if d == 0:
            return None
        dy = L.p[1] - Seg.p[1]
        dx = L.p[0] - Seg.p[0]
        ua = (Seg.v[0] * dy - Seg.v[1] * dx) / d
        #In pyeuclid, _u_in() for LineSegment2 says:
        # return (u >= 0.0 and u <= 1.0)
        if not L._u_in(ua):
            return None
        ub = (L.v[0] * dy - L.v[1] * dx) / d
        if not Seg._u_in(ub):
            return None
        return V( [L.p[0] + ua * L.v[0], L.p[1] + ua * L.v[1] ])
        ##########################################################

def normalize(v):
    d = sqrt(v[0] ** 2 + \
             v[1] ** 2 )
    if d:
        v[0] /= d
        v[1] /= d
    return v

def distance(p, q):
    c = LineSegment2(p, q)
    if c:
        return sqrt(c.v[0] ** 2 + \
                    c.v[1] ** 2 )
    else: return 0.0

def moved(p, v):
    #return (p[0] + v[0], p[1] + v[1])
```

```
        for p in self.gridPts:
            gobj += circle([p[0], p[1]], 6, color="grey")

        return gobj

def placeMihrab(site):
    gobj = Graphics()
    c = centroid(site.vl)
    gobj += circle(c, 4)
    lnQiblah = Line2(c, site.Q)
    spine_ends = lnIsnPoly(lnQiblah, site.vl)
    front = spine_ends[1]
    back  = spine_ends[0]
    sv = (front - back) #spine vector
    spine = line2d([back, front], thickness=2, color="red")
    gobj += spine
    #place Mihrab heuristically 1/5 of the way back down spine
    mhbPt = moved(back, V([sv[0]*4/5, sv[1]*4/5]))
    #In unit circle, Sin s = s.y/1, Cos s = s.x/1
    #sA = atan(sv[1]/sv[0])
    endsQiblahWallLn = lnIsnPoly( Line2(mhbPt, site.Q90), site.vl)
    lenQiblahWall = fracQW * distance(endsQiblahWallLn[0], endsQiblahWallLn
[1])
    mphTLcnr = moved(mhbPt, V(site.Q270) * lenQiblahWall/2)
    wallQiblah = LineSegment2( mphTLcnr, V(site.Q90) * lenQiblahWall )
    mhb = circle(mhbPt, 12)
    gobj += mhb
    #now draw Qiblah Wall
    gobj += line2d([mphTLcnr, moved(mphTLcnr, wallQiblah.v)], thickness=2,
color="black")
    return gobj


#__main__ starts here
V = VectorSpace(RR, 2)
q = normalize(V([-1.0, 2.0]))  # Qiblah direction
n = normalize(V([-sqrt(2.0), -sqrt(2.0)])) # True North
c = normalize(V([0, -1.0]))  #gav City Grid "N" dirn. == up on screen or
err!

site = Site(q, c, n, [])
fracQW = 0.80
graphic = Graphics()
graphic += frameCanvas()
graphic += site.drawSite()
graphic += placeMihrab(site)
graphic += site.drawGrid()

graphic.show(figsize = [12, 9], axes=False)
```

```
        gridIntsns(self.gridPts, self.vl, self.Q, self.grid['q'])
        #Place CG axes at each intsn pt. in gridPts
        gridIntsns(self.gridPts, self.vl, self.C, self.grid['c'])
        # ###############################################
        #Now find intersections of C.x , Q.y  &  C.y , Q.x
        #using my intersection function for line segements
        for sQy in self.grid['q']['y']:
            for sCx in self.grid['c']['x']:
                intsn = sQy.intersect(sCx)
                if intsn:
                    self.gridPts.append(intsn)
        for sQx in self.grid['q']['x']:
            for sCy in self.grid['c']['y']:
                intsn = sQx.intersect(sCy)
                if intsn:
                    self.gridPts.append(intsn)

        #Now find intersections and add to gridPts
        #Place QG axes at each intsn pt. in gridPts
        gridIntsns(self.gridPts, self.vl, self.Q, self.grid['q'])
        #Place CG axes at each intsn pt. in gridPts
        gridIntsns(self.gridPts, self.vl, self.C, self.grid['c'])


        #Draw each grid line we've calculated
        #Qiblah Grid #clr = "#DD4444"
        for s in self.grid['q']['y']:
            gobj += line2d([s.p, moved(s.p, s.v)], thickness = 1,
color="red")
            #gridln = GridLn(blah, blah)
            #DBSession.add(gridln)
        for s in self.grid['q']['x']:
            gobj += line2d([s.p, moved(s.p, s.v)], thickness = 1,
color="red")
        #City Grid   #clr = "#3355BB"
        for s in self.grid['c']['y']:
            gobj += line2d([s.p, moved(s.p, s.v)], thickness = 1,
color="blue")
        for s in self.grid['c']['x']:
            gobj += line2d([s.p, moved(s.p, s.v)], thickness = 1,
color="blue")
        #True North  #clr = "#33BB55"
        for s in self.grid['n']['y']:
            gobj += line2d([s.p, moved(s.p, s.v)], thickness = 1,
color="green")
        for s in self.grid['n']['x']:
            gobj += line2d([s.p, moved(s.p, s.v)], thickness = 1,
color="green")

        #Circle each grid point we've found.
```

```
    return p + v #or V(p) + V(v)

def translate(pl, v):
    return map( (lambda p: V([ p[0]+v[0], p[1]+v[1]]) ), pl )
```

```
''' Created on 14/09/2010
@author: munrog '''

import sys
from sage.all import *
from sage.plot.plot3d.shapes2 import Line  #as Line2
from sage.plot.plot3d.shapes2 import Point #as Point2
from sage.plot.polygon import polygon2d

V = VectorSpace(RR, 2)

#library utilities ##################################

def pIntsnLS(P, L):
    #find intsn Point P.x, P.y and LS(p,v)
    if L.p == P: #projn will be 0
        return L.p
    else:
        diffv = P - L.p
        #theta = arccos(diffv.dot(L.v) / diffv.magnitude()*Lv.magnitude())
        #theta = 0 iff cos(theta) = 1 <=> ratio of magn^2 = 1
        d2 = diffv.magnitude_squared()
        s2 = L.v.magnitude_squared()
        ratio = diffv.dot(L.v) / (d2*s2) #arccos diffv/d*s gives theta
        #gav Square root used to calc magnitude is an expensive operation
        if ratio == 1:
            return P
        else:
            return None

def edgeSeq(pLoop):
    eLoop = []
    for j in range(len(pLoop)):
        vertex = V(pLoop[j]) #vectorize tuple?
        next = V(pLoop[(j+1)%len(pLoop)])
        #assert isinstance((next-vertex), Vector2)
        e = LineSegment2( vertex, next - vertex )
        eLoop.append(e)
    return eLoop

def areaPoly(polygon):
    area = 0
    n = len(polygon)
    for i in range(n):
        j = (i+1) % n
        area = area + polygon[i][0] * polygon[j][1]
        area = area - polygon[i][1] * polygon[j][0]
    area = area / 2
    return abs(area)
```

```
def centroid(polygon):
    n = len(polygon)
    A = areaPoly(polygon)
    p = polygon
    factor = 0
    cx = 0
    cy = 0
    for i in range(n):
        j = (i+1) % n
        factor = p[i][0] * p[j][1] - p[j][0] * p[i][1]
        cx = cx + (p[i][0] + p[j][0]) * factor
        cy = cy + (p[i][1] + p[j][1]) * factor
    A *= 6.0
    factor = 1/A
    cx *= factor
    cy *= factor
    return (cx, cy)

def lnIsnPoly(line, polygon):
    ends = [] #may be one or two points(for convex)
    edges = edgeSeq(polygon)
    first_edge = edges[0]
    #last pt in polygon occurs also in first edge
    pt0 = first_edge.p #isOn(first_edge.p, line) #gav get rid?
    #poly = polygon2d([e.p for e in edges]) #dbg
    for edge in edges:
        intsn = line.isnLS(edge)
        if intsn:
            ends.append(intsn)
    if len(ends) == 3:
        if ends[0][0] == ends[1][0] and ends[0][1] == ends[1][1]:
            return [ends[1], ends[2]]
        elif ends[1][0] == ends[2][0] and ends[1][1] == ends[2][1]:
            return [ends[0], ends[1]]
        elif ends[0][0] == ends[2][0] and ends[0][1] == ends[2][1]:
            return [ends[0], ends[1]]
    elif len(ends) == 2:
        if ends[0][0] == ends[1][0] and ends[0][1] == ends[1][1]:
            return [ends[0]] #they're the same point
        else:
            return ends
    else:
        return None


#end library of utilities ###################################
```

```
    def drawSite(self):
        gobj = Graphics()
        for p in self.vl:
            seg = line2d([p, self.vl[(self.vl.index(p)+1)%len(self.vl)]],
thickness=2, color="black")
            gobj += seg
        return gobj

    def drawGrid(self):
        #This site will have a grid
        gobj = Graphics()

        def gridIntsns(ptList, site, Vec, gridLines):
            #gridLines = { 'x':[], 'y':[] }
            for p in ptList:
                Ly = Line2(p, Vec)
                Lx = Line2(p, V([Vec[1], -Vec[0]])) #ie. normal to Vec
                yEnds = lnIsnPoly(Ly, site)
                if yEnds and len(yEnds) == 2:
                #Ignore everything outside site bounds
                    ySeg = LineSegment2(yEnds[0], yEnds[1] - yEnds[0])
                    gridLines['x'].append(ySeg)
                xEnds = lnIsnPoly(Lx, site)
                if xEnds and len(xEnds) == 2:
                    xSeg = LineSegment2(xEnds[0], xEnds[1] - xEnds[0])
                    gridLines['y'].append(xSeg)

        #place QG axes at each corner of self.vl
        gridIntsns(self.vl, self.vl, self.Q, self.grid['q'])
        #place CG axes at each corner of self.vl
        gridIntsns(self.vl, self.vl, self.C, self.grid['c'])
        #place TN axes at each corner of self.vl
        gridIntsns(self.vl, drawing_area, self.N, self.grid['n'])


        #Now find intersections of C.x , Q.y  &  C.y , Q.x
        #using my intersection function for line segements
        for sQy in self.grid['q']['y']:
            for sCx in self.grid['c']['x']:
                intsn = sQy.intersect(sCx)
                if intsn:
                    self.gridPts.append(intsn)
        for sQx in self.grid['q']['x']:
            for sCy in self.grid['c']['y']:
                intsn = sQx.intersect(sCy)
                if intsn:
                    self.gridPts.append(intsn)

        #Now find intersections and add to gridPts
        #Place QG axes at each intsn pt. in gridPts
```

```
from sage.all import * #Sage authors prefer this
from sage.plot.polygon import polygon2d
from sage.plot.plot import Graphics

load('/home/bee/Downloads/util2.py')
load('/home/bee/Downloads/vectors.py')

w = 1024
h = 768

V = VectorSpace(RR, 2) #This defines in Sage a 2D vector space over the
Real Numbers
O = V([0, 0]) #The Origin


drawing_area = [V([10,10]), V([w-10, 10]), V([w-10, h-10]), V([10, h-10])]

def frameCanvas():
    gobj = Graphics()
    #fill window with grey background
    rec1 = polygon2d( [(0,0), (w, 0), (w, h), (0, h)], rgbcolor=(1/2, 1/2,
1/2) )
    #Canvas is a white rectangle w/ 10 pixel border
    rec2 = polygon2d( drawing_area, rgbcolor=(1, 1, 1))
    gobj += rec1
    gobj += rec2
    return gobj

class Site():
    #The site is a seq of Points for polygonal site bounds including
setbacks
    #site = [Point2(381, 44), Point2(w-360, 44), Point2(w-50, 200),
    #        Point2(w-h/2+70, h-40), Point2(h/2-50, h-40), Point2(40, w-h/
2-75)]

    def __init__(self, q, c, n, vl):
        self.vl = [ V([80, 100]), V([w-120, 40]), V([w-120, h-360]), V
([w-500, h-40]), V([40, h-40]) ]
        self.Q = q
        self.C = c
        self.N = n
        self.siteBounds = edgeSeq(self.vl)
        self.gridLines = []
        self.gridPts = []
        self.Q90 = (-q[1], q[0])
        self.Q270 = (q[1], -q[0])
        self.grid = { 'q':{'x':[], 'y':[]}, 'c':{'x':[], 'y':[]}, 'n':{'x':
[], 'y':[]} }
```

7.6 Appendix F

Mosque Room List

Table 7-1 Mosque Room List

	FUNCTIONS	CODE	Capacity/No's	REMARKS
1	Outside-Front	F		
2	Outside-Right	R		
3	Outside-Left	L		
4	Outside-Back	B		
	Prayer Halls Zone	PHZ		
	Main Prayer Hall	MPH		
5	Main Warship Area	MWA	2500 prayers	
6	Mihrab	MHB	1	In the centre of the Qiblah wall
7	Minbar	MBR	1	Left side of the Mihrab, 1.5m above floor level
8	Qura'an Stands	QS	2000 books	Distributed all over the prayer hall
	Main Prayer Hall Entrances	MPHE	4	Imam, Main, left, Right
9	Imam Entrance Door	IED	1 door	At the Qiblah wall
10	Imam Entrance Foyer	IEF	1	Connects the entrance with the prayer hall and the Minbar
11	Prayer Hall Main Doors	PHMD	3 doors	Back side wall of the prayer hall
12	Main Entrance transition Area	META	1	Inside of the prayer hall by the doors
13	Main Entrance Shoes Racks Area	MESRA	1000 shoes	Outside of the prayer hall by the doors
14	Prayer Hall Right Side Doors	PHRD	2 doors	Right side wall of the prayer hall
15	Right-Side Entrance transition Area	RETA	1	Inside of the prayer hall by the doors
16	Right-Side Entrance Shoes Racks Area	RESRA	750 shoes	Outside of the prayer hall by the doors
17	Prayer Hall Left Side Doors	PHLD	1	Left side wall of the prayer hall
18	Left-Side Entrance transition Area	LETA	2 doors	Inside of the prayer hall by the doors
19	Left-Side Entrance Shoes Racks Area	LESRA	750 shoes	Outside of the prayer hall by the doors
	Female Prayer Hall	FPH		
20	Female Worship Area	FWA	500 prayers	Preferably located in a mezzanine level
21	Qura'an Stands	QS	400 books	Distributed all over the prayer hall
	Females Prayer Hall Entrances	FPHE	2	
22	Females Prayer Hall Doors	FPHED	2	Accessible from the courtyard
23	Females Entrance Foyer	FEF	2	Connects the entrance with

				the stairs and the females ablutions
24	Female Shoes Racks Area	FSRA	2X250 shoes	Inside the foyer
25	Females Prayer Hall Stairs	FPHS	2	Connects the foyer to the prayer hall
	Courtyard Zone	CYZ	1000 sq. m	40% of the main prayer hall
26	Courtyard Open Area	CYOA	600 sq. m	
	Porticos	PRT		
27	Portico Qiblah Side	PQ	300 prayer	Located at the backside of the prayer hall
28	Portico Right Side	PR	1	Right Side arcade
29	Portico Left Side	PL	1	Left Side arcade
30	Portico Back Side	PB	1	Back Side arcade
	Mosque Entrances	ME		Gates to the courtyard through the surrounding arcades
31	Mosque Main Entrance (Back)	MMEB	1	Gate at the Back Side arcade
32	Mosque Side Entrance (Right)	MMER	1	Gate at the Right Side arcade
33	Mosque Side Entrance (left)	MMEL	1	Gate at the Left Side arcade
	Ancillaries Zone	AZ		
34	Crpet Storage Room	CSR	1	250 m long, 1.5 m width Rolls
35	Prayer Hall Janitorial Room	PHJR	1	
	Males Ablutions	MA		
36	Males Ablution Area	MAA	100 ablutions	1:25 prayers
37	Males Toilets (WC's)	MTWC	25 cubicles	1:100 prayers
38	Males Toilets (Basins)	MTB	25 basins	1:100 prayers
39	Males Toilet Janitorial Room	MTJR	1	
	Females Ablution	FA		
40	Females Ablution Area	FAA	20 ablutions	1:25 prayers
41	Females Toilets (WC's)	FTWC	5 cubicles	1:100 prayers
42	Females Toilets (Basins)	FTB	5 basins	1:100 prayers
43	Females Toilet Janitorial Room	FTJA	1	
	Madrasa	MDS		
44	Library	LIB	1	Books stack area, reading area and meeting table
45	Classroom	CR	2	Each, 22 students desks and chairs, teacher desk and chair and book cabinet
46	Office	OFC	1	Desk, desk chair, 2 visitors

				chairs, coffee tables, book shelves, file cabinets, meeting table, single sofa chair, double sofa and a cocktail table
47	Reception	RCP	1	Counter, chair, file cabinet, shelves, lockers
	Services	SRV	3	
48	General storage Room	GSR	1	
49	Mechanical Room	MCH	1	
50	Electrical Room	ELT	1	
51	Care Taker Suite	CTS	1	
52	Care Taker Bedroom	CTBR	1	Single bed, night stand, wardrobe and a chair
53	Care Taker Kitchenette	CTK	1	Fridge, stove, sink, counter and cabinets
54	Care Taker Living	CTL	1	2 chairs dining table, double sofa, cocktail table TV stand, book shelves
55	Care taker Toilet	CTT	1	Basin, WC, stand shower, mirror
56	Minaret	MNT	2	30 m high
57	Car Parking	CP	80 cars	1:25 Prayer

7.7 Appendix G

GDMW Standard for a mosque design project

3000-MAN MOSQUE
STANDARDIZATION
DESIGN CRITERIA

STRUCTURAL

01 FOUNDATIONS

011 STANDARD FOUNDATIONS
Individual isolated footing

02 SUBSTRUCTURE

021 SLAB ON GRADE & BEAM
120mm slab-on-grade

03 SUPERSTRUCTURE

031 UPPER FLOOR COLUMNS & DECKS
Drop beam, solid slab

032 ROOF COLUMNS & DECKS
Drop beam, solid slab, and high roof steel structure

ARCHITECTURAL

04 EXTERIOR CLOSURE

041 EXTERIOR WALL ASSEMBLIES
For mosque, use cavity wall (10-5-15) 100mm thk CMU outer wall, 50mm thk rigid insulation, 150mm thk inner wall
For ablutions/toilets & minaret, use 200mm thk CMU exterior wall

042 EXTERIOR DOORS & WINDOWS

For Mosque, use:
- Double-glazed anodized alum. sliding window for ground floor windows
- Double-glazed anodized alum. fixed windows for mezzanine windows
- Double-glazed anodized alum. casement windows for toilets
- Decorative metal door w/ side lights/transom for main praying area
- Hollow metal doors for other exterior doors

For ablution/toilets, use:
- Single glazed anodized alum. casement windows
- Hollow metal doors for toilet exterior doors

05 ROOFING

For mosque, use:
- Standard built-up roofing but use PU system (polyurethane foam) as thermal insulation and waterproofing & pea gravel topping
- In the central core over the open praying hall, use steel frame + sandwich insulated roofing panels

For ablution/toilet & minaret, use:
- Standard built-up roofing but use PU system (polyurethane foam) as thermal insulation and waterproofing & pea gravel topping

06 INTERIOR CONSTRUCTION

0611 INTERIOR WALLS & PARTITIONS

For interior wall partitions, use 200mm thk CMU except in toilet cubicles that is 100mm thk CMU.

0612 INTERIOR DOORS & WINDOWS

For interior doors, use hollow metal doors & frames
For toilet cubicles, use aluminum doors
For daily praying, use hollow core wood folding doors
For mezzanine praying, use single glaze clear glass on GRC lattice works

0621 INTERIOR FLOOR FINISHES

For praying hall, use
- 60m wide side border unglazed ceramic
- Pre-cut made praying carpet 3.90 wide (1.30 x 3 rows) over plain concrete flooring

For arcade use granite floor tiles
For sahan use interlocking paving tiles with patterns
For stairs & landings, use terrazzo slabs & terrazzo tiles
For ablution & toilets, use ceramic tiles
For ramps, use plain concrete

0622 INTERIOR WALL FINISHES

For mosque, use plaster & paint
For toilets & ablution, use glazed ceramic, door height extent
For minaret, fairface without paint

0621 INTERIOR CEILING FINISHES

For mosque, use acoustic tile
For central core & dome, use gypsum board paint finish
Fro toilets/ablution, use fairface paint finish

MECHANICAL

08 MECHANICAL SYSTEMS

0811 PLUMBING WATER SUPPLY

From an underground water tank, water is transferred to the overheads and from there it is gravity system

0811 PLUMBING – SEWER & STORM

Gravity system, PVC pipes to manhole up to the sewer tapping point

0821 HVAC – CENTRAL EQUIPMENT

Package units ducted for all praying halls, split units for ladies praying hall & Imam

0821 HVAC - DISTRIBUTION

Office/Library. All praying hall with exhaust system. For the unoccupied area during ordinary days, use desert cooler.

083 FIRE PROTECTION

Use fire extinguishers only

ELECTRICAL

09 ELECTRICAL SYSTEMS

Use fluorescent fixtures 3x36 or 4x36 watts for the main prayer hall area. Use comp. Fluorescent downlights at entrances and wall bracket lights at the sahan area. Provide cove lighting around the dome with fluorescent tubes. Provide metal halide downlights at the central high-level ceiling and in the dome area. Average illuminance level recommended in the main prayer hall area is 250 to 275 lux. All the lighting fixtures in the staircase shall have battery backup for one of the lamps and are rated for 30 minutes.

Provide 220 and 127 V general-purpose receptacles in the prayer hall and in the Imam's office.

Public address system shall include for ceiling type speakers in the hall area and external speakers on the Minaret. At the minbar for Friday prayer and at the minbar for daily prayer provide microphone outlets and power control switches for central amplifier.

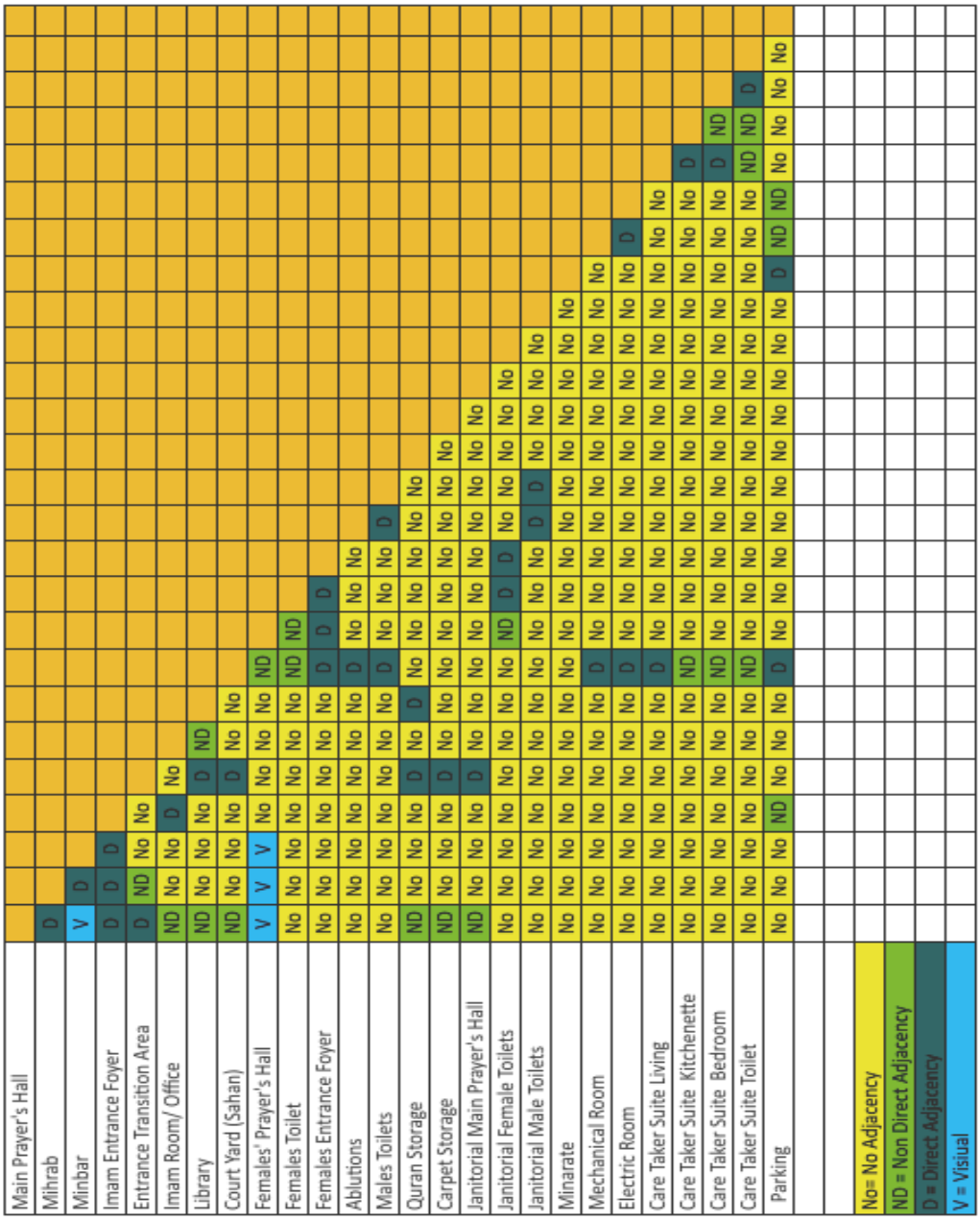
Provide fire alarm system, zonal type.
Provide a telephone outlet in the Imam's office/library.
Provide lightning protection system and grounding system.

7.8 Appendix H

Adjacency matrix

An example of adjacency matrix developed to govern the spatial relationships between the chosen project's rooms.

Table 7-2 Adjacency Matrix



References

Aazam, Z. (2007). The social logic of the mosque: A study in building typology. Paper presented at the *6th International Space Syntax Symposium*. Istanbul: Istanbul Technical University Faculty of Architecture. from http://www.spacesyntaxistanbul.itu.edu.tr/

Adams, J. L. (1991). *Flying buttresses, entropy, and O-rings: The world of an engineer*. Cambridge, Mass: Harvard University Press.

Agarwal, M., Cagan, J., & Constantine, K. G. (1999). Influencing generative design through continuous evaluation: Associating costs with the coffeemaker shape grammar. *Artificial Intelligence For Engineering Design Analysis & Manufacturing, 13*(4), 253–275.

Agkathidis, A. (2009). *Modular structures in design and architecture*. Amsterdam: Bispublishers.

Akin, O. (1986). *Psychology of architectural design*. London: Pion.

Akin, O. (1990). Necessary conditions for design expertise and creativity. *Design Studies, 11*(2), 107–113.

Akin, O., & Moustapha, H. (2004). Strategic use of representation in architectural massing. *Design Studies, 25*(1), 31–50.

Aksamija, A. (2007). The Generic Mosque. Design Principles for Mosque Design. *Critical Design* (24), 51–61. Retrieved 22/04/2009, from http://tdd.elisava.net/coleccion/24/aksamija-en/view?set_language=en

Al-Madhi, F. (n/d). Design Criteria for Mosques. Riyadh, Saudi Arabia: General Directorate of Military Works (GDMW).

Alexander, C. (1964). *Notes on the synthesis of form*. Cambridge, Mass: Harvard University Press.

Alexander, C. (1971). The state of the art in design methodology (Questions by M. Jacobson). *DMG Newsletter,* pp. 3–7.

Alexander, C., Ishikawa, S., & Silverstein, M. (1977). *A pattern language: Towns, buildings, construction*. New York: Oxford University Press.

Alexanderson, G. L. (2006). About The Cover: Euler And Königsberg's Bridges: A Historical View. *American Mathematical Society. Bulletin, New Series, of the American Mathematical Society, 43*(4), p. 7.

Allam, A. (2002). Understanding Islam, its birth and ascent. *Junior Scholastic, 105*(8), p. 10.

The American Heritage Science Dictionary (2005). Boston: Houghton Mifflin Co.

Aranda, B., & Lasch, C. (2006). *Tooling*. New York: Princeton Architectural Press.

Arbib, M. A. (2003). *The handbook of brain theory and neural networks* (2nd ed.). Cambridge, Mass: MIT Press.

Ardalan, N. (1980). The Visual Language of Symbolic Form: A Preliminary Study of Mosque Architecture. *Architectural Transformations in the Islamic World* Fez, Morocco: Aga Khan Award for Architecture.

Augenbroe, G. (2002). Trends in building simulation. *Building and Environment, 37* (8-9), 891–902.

Bachman, L. R. (2003). *Integrated buildings: The systems basis of architecture*. Hoboken, NJ: John Wiley & Sons.

Baer, A., Eastman, C., & Henrion, M. (1979). Geometric modelling: A survey. *Computer-Aided Design, 11*(5), 253–272.

Bartlett, F. C. S. (1995). *Remembering: A study in experimental and social psychology*. Cambridge, New York: Cambridge University Press. (Original work published 1932).

Bertel, S., Freksa, C., & Vrachliotis, G. (2004). Aspectualize and Conquer in Architectural Design. In J. Gero, B. Tversky & T. Knight (Eds.), *Visual and Spatial Reasoning in Design III* (pp. 255–279): Key Centre of Design Computing and Cognition, University of Sydney

Bhatt, M., Dylla, F., & Hois, J. (2009). Spatio-terminological Inference for the Design of Ambient Environments. In K. Hornsby, C. Claramunt, M. Denis & G. Ligozat (Eds.), *Spatial Information Theory* (Vol. 5756, pp. 371–391): Springer Berlin / Heidelberg.

Bhatt, M., Hois, J., Kutz, O., & Dylla, F. (2010). Modelling Functional Requirements in Spatial Design. Paper presented at the *29th International Conference on Conceptual Modeling (ER2010)*. Vancouver, BC, Canada: Springer.

Biggs, N., Lloyd, E. K., & Wilson, R. J. (1986). *Graph theory, 1736–1936 / Norman L. Biggs, E. Keith Lloyd, Robin J. Wilson*. Oxford [Oxfordshire]; New York: Clarendon Press.

Blessing, L. T. M., & Chakrabarti, A. (2009). *DRM, a Design Research Methodology*. London: Springer.

Boddy, S., Rezgui, Y., Cooper, G., & Wetherill, M. (2007). Computer integrated construction: A review and proposals for future direction. *Advances in Engineering Software, 38*(10), 677–687.

Broadbent, G. (1981). The Morality of Designing. *Jacques and Powell*, 309–328.

Broadbent, G. (1988). *Design in architecture: architecture and the human sciences*. London, UK: David Fulton Publishers Ltd. (Original work published 1973 John Wiley & Sons, New York, USA).

Buchanan, R. (2004). Design as inquiry: The common, future and current ground of design. Paper presented at the *Futureground: Design Research Society International Conference 2004, 17–21 November 2004, Monash University, Faculty of Art & Design*. Melbourne, Australia: Monash University.

Buck, L. V. (1915). *Geometric patterns as the basis of design.* University of California, Berkeley. Retrieved from hdl.handle.net database.

Caldas, L. (2008). Generation of energy-efficient architecture solutions applying GENE_ARCH: An evolution-based generative design system. *Advanced Engineering Informatics, 22*(1), 59–70.

Castéra, J.-M., Peuriot, F., & Ploquin, P. (1999). *Arabesques: Decorative art in Morocco* (K. McElhearn, Trans.). Courbevoie (Paris): ACR Edition Internationale

Central Intelligence Agency. (2010). *The World Fact book* Washington, DC Central Intelligence Agency https://http://www.cia.gov/library/publications/the-world-factbook/index.html. Retrieved 2010

Chan, K. H. (2008). *A computational kernel for supporting generative and evolutionary design.* Unpublished Ph.D., Hong Kong Polytechnic University, Hong Kong. Retrieved from ProQuest Dissertations & Theses (PQDT) database.

Chase, S. C. (2005). Generative design tools for novice designers: Issues for selection. *Automation in Construction, 14*(6), 689–698.

Chynoweth, P. (2004). Progressing the rights to light debate: Part 1: A review of current practice. *Structural Survey, 22*(3), p. 131.

Chynoweth, P. (2005). Progressing the rights to light debate: Part 2: The grumble point revisited. *Structural Survey, 23*(4), p. 251.

Chynoweth, P. (2009). Progressing the rights to light debate: Part 3: Judicial attitudes to current practice. *Structural Survey, 27*(1), p. 7.

Clarke, J. A. (2001). *Energy simulation in building design* (2nd ed.). Oxford: Butterworth Heinemann.

Clocksin, W. F., & Mellish, C. S. (2003). *Programming in Prolog* (5th ed.). Berlin New York: Springer-Verlag.

Crawley, D. B., Lawrie, L. K., Pedersen, C. O., & Winkelmann, F. C. (2000). EnergyPlus: Energy simulation program. *ASHRAE Journal, 42*(4), p. 49.

Cross, N. (1982). Designerly ways of knowing. *Design Studies, 3*(4), 221–227.

Cross, N. (1993). Science and design methodology: A review. *Research in Engineering Design, 5*(2), 63–69.

Cross, N. (1999). Natural intelligence in design. *Design Studies, 20*(1), 25–39.

Cross, N. (2001). Designerly ways of knowing: Design discipline versus design science. *Design Issues, 17*(3), 49–55.

Cross, N. (2004). Expertise in design: An overview. *Design Studies, 25*(5), 427–441.

Cross, N. (2007). Forty years of design research. *Design Studies, 28*(1), 1–4.

Cross, N., Naughton, J., & Walker, D. (1981). Design method and scientific method. *Design Studies, 2*(4), 195–201.

Csikszentmihalyi, M. (1988). Society, culture, and person: A systems view of creativity. In R. J. Sternberg (Ed.), *The nature of creativity: Contemporary psychological perspectives.* (pp. 325–339). New York, NY US: Cambridge University Press.

Csikszentmihalyi, M. (1996). *Creativity: Flow and the psychology of discovery and invention* (1st ed.). New York: HarperCollinsPublishers.

Csikszentmihalyi, M., & Getzels, J. W. (1971). Discovery-oriented behavior and the originality of creative products: A study with artists (Vol. 19, pp. 47–52).

Darke, J. (1979). The primary generator and the design process. *Design Studies, 1*(1), 36–44.

Day, L. F. (1999). *Pattern design*. Mineola, NY: Dover Publ.

De Bono, E. (1971). *The use of lateral thinking*. Harmondsworth: Penguin.

De Chiara, J., & Callender, J. H. (1980). *Time-saver standards for building types* (2d ed.). New York: McGraw-Hill.

De Landa, M. (2002). Deleuze and the use of the genetic algorithm in architecture [Article]. *Architectural Design*(155), 9–12.

Dodd, N. (1990). Optimisation of network structure using genetic techniques *International Joint Conference on Neural Networks (IJCNN)* (pp. 965–970).

Dollens, D. (2006). The Cathedral Is Alive: Animating Biomimetic Architecture. *Animation, 1*(1), 105–117.

Drake, J. (1979). The Primary Generator and the Design Process. *Design Studies, 1*(1), 36–44.

Ferguson, E. S. (1977). The mind's eye: Nonverbal thought in technology. *Science, 197*(4306), 827–836.

Ferguson, E. S. (1992). *Engineering and the mind's eye*. Cambridge, Mass: MIT Press.

Flanagan, R. H. (2005). Generative logic in digital design. *Automation in Construction, 14*(2), 241–251.

Frampton, K. (1995). *Studies in tectonic culture : the poetics of construction in nineteenth and twentieth century architecture*. Cambridge, Mass: Graham Foundation for Advanced Studies in the Fine Arts. MIT Press.

Frazer, J. (1995). *An Evolutionary Architecture*. London, UK: Architectural Association Publications.

Frazer, J. (2002). Creative design and the generative evolutionary paradigm. In D. Corne & P. Bentley (Eds.), *Creative evolutionary systems* (pp. 253–274). California, San Francisco: Morgan Kaufmann Publishers Inc. (Elsevier).

Frazer, J. H. (1974). Reptiles. *Architectural Design*, 231-239.

Frazer, J. H., & Connor, J. M. (1979). A Conceptual Seeding Technique for Architectural Design. *Proceedings of International Conference*

on the Application of Computers in Architectural Design (pp. 425–434). Berlin: Online Conferences with AMK.

Frishman, M., Khan, H.-U., & Al-Asad, M. (1994). *The mosque: History, architectural development & regional diversity*. New York: Thames and Hudson; c1994.

Gallaher, M. P., O'Connor, A. C., Dettbarn Jr, J. L., & Gilday, L. T. (2004). Cost Analysis of Inadequate Interoperability in the U.S. Capital Facilities Industry: National Institute of Standards and Technology (NIST).

Galle, P. (1981). An algorithm for exhaustive generation of building floor plans. *Commun. ACM, 24*(12), 813–825.

Galle, P. (2008). Candidate worldviews for design theory. *Design Studies, 29*(3), 267–303.

Glanville, R. (1998). Researching design and designing research. *Strandman, P., editor, No Guru, No Method? Discussion on Art and Design Research*, 55.

Goldberg, D. E. (1989). *Genetic algorithms in search, optimization, and machine learning*. Reading, Mass.: Addison-Wesley Pub. Co.

Grason, J. (1968). A Dual Linear Graph Representation for Space-Filling Location Problems of the Floor Plan Type. Paper presented at the *The Design Methods Group First International Conference*. Cambridge, Massachusetts: MIT Press.

Grason, J. (1970). A dual linear graph representation for space-filling location problems of the floor plan type. In G. T. Moore (Ed.), *Emerging Methods in Environmental Design and Planning* (pp. 170–178). Cambridge: MIT Press.

Grason, J. (1971). An approach to computerized space planning using graph theory. Paper presented at the *Proceedings of the 8th Design Automation Workshop*. Atlantic City, New Jersey, United States: ACM.

Greenberg, I. (2007). *Processing : creative coding and computational art*. Berkeley, Calif.: Friendsof.

Gruau, F. (1993). Cellular encoding as a graph grammar. *IEE Colloquium on Grammatical Inference: Theory, Applications and Alternatives* (pp. 17/11–17/10). Colchester, UK.

Gu, Z., Tang, M. X., & Frazer, J. H. (2006). Capturing aesthetic intention during interactive evolution. *Computer Aided Design, 38*(3), 224–237.

Guo, R.-Y., Wong, S. C., Huang, H. J., Zhang, P., & Lam, W. H. K. (2010). A microscopic pedestrian simulation model and its application to intersecting flows. *Physica A: Statistical Mechanics and its Applications, 389*(3), 515–526.

Haji Mohamad Rasdi, M. T. (1998). *The mosque as a community development centre: Programme and architectural design guidelines for contemporary Muslim societies*. Skudai, Johor Darul Ta'zim: Penerbit Universiti Teknologi Malaysia.

Harris, D. J. (2009). Design theories and cognitive science: The embodied mind and design. In J. Dehlinger & J.-P. Protzen (Eds.), *Architecture - design methods - Inca structures Festschrift for Jean-Pierre Protzen* (pp. 94–103). Kassel: Kassel Univ. Press.

Heidegger, M. (1971). The Origin of the Work of Art. In S. M. Cahn & A. Meskin (Eds.), *(2007) Aesthetics : a comprehensive anthology* (pp. 344–357). Malden, MA: Blackwell Pub.

Herr, C. M., & Kvan, T. (2007). Adapting cellular automata to support the architectural design process. *Automation in Construction, 16*(1), 61–69.

Hilali, T. a.-D., & Khan, M. M. (2002). *Interpretation of the meanings of the noble Qur'an in the English language*. Riyadh, Saudi Arabia: Darussalam.

Hillier, B., Musgrove, J., & O'Sullivan, P. (1972). Knowledge and design. Paper presented at the *The 3rd. annual Environmental Design Research Association Conference 1972*. Los Angeles, CA, USA: American Institute of Architects.

Hirsch, E. D., Kett, J. F., & Trefil, J. S. (2002). *The new dictionary of cultural literacy.*http://www.credoreference.com/vol/221

Holland, J. H. (1992). *Adaptation in natural and artificial systems : an introductory analysis with applications to biology, control, and artificial intelligence* (1st ed.). Cambridge, Mass: MIT Press.

Holness, G. V. R. (2006). Building Information Modeling. *ASHRAE Journal, 48*(8), 38–46.

Holzer, D., Hough, R., & Burry, M. (2007). Parametric Design and Structural Optimisation for Early Design Exploration. *International Journal of Architectural Computing, 5* (4), 625–643.

Homayouni, H. (2007). *A Genetic Algorithm Approach to Space Layout Planning Optimization.* Unpublished Master Thesis, University of Washington, Washington.

Hong, T., Chou, S. K., & Bong, T. Y. (2000). Building simulation: An overview of developments and information sources. *Building and Environment, 35*(4), 347–361.

Hoogendoorn, S. P., & Daamen, W. (2005). Pedestrian Behavior at Bottlenecks. *Transportation Science, 39*(2), 147.

Howard, G. S. (1983). Toward methodological pluralism. *Journal of Counseling Psychology, 30*(1), 19–21. from pdh database.

Hübsch, H. (1992). *In what style should we build? The German debate on architectural style / Heinrich Hübsch ... [et al.]; introduction and translation by Wolfgang Herrmann*. Santa Monica, CA: Getty Center for the History of Art and the Humanities; [Chicago] : Distributed by the University of Chicago Press.

Hussaini, M. Y., Alexandrov, N. M., Institute for Computer Applications in Science and Engineering, Langley Research Center, & ICASE/NASA Langley Workshop on Multidisciplinary Design (1997). *Multidisciplinary design optimization: state of the art/edited by Natalia M. Alexandrov, M.Y. Hussaini*. Philadelphia: Society for Industrial and Applied Mathematics.

Ibn Kathir, I. i. U. (2003). *Tafsir ibn Kathir: (abridged)* (S. A. Mubarakfuri, Trans.). Riyadh: Darussalam.

Ibrahim, H. M. (1979). *Planning Standards for Mosques*. Riyadh, Saudi Arabia: Ministry of Municipal and Rural Affairs and the Development Program of the United Nations.

Janssen, P., Frazer, J., & Tang, M. x. (2002). Evolutionary Design Systems and Generative Processes. *The International Journal of Artificial Intelligence, Neural Networks, and Complex Problem Solving Technologies, 16*(2), 119–128.

Janssen, P. H. T. (2004). *A design method and computational architecture for generating and evolving building designs.* Unpublished Ph.D. Thesis, Hong Kong Polytechnic University (People's Republic of China), Peoples Republic of China. Retrieved from ProQuest Information and Learning Company database.

Järvinen, P. (2000a). On a variety of research output types. In L. Svensson, U. Snis, C. Sorensen, H. Fägerlind, T. Lindroth, M. Magnusson & C. Östlund (Eds.), *IRIS23. Laboratorium for Interaction* (pp. 251–265). Sweden: University of Trollhattan Uddevalla.

Järvinen, P. (2000b). Research questions guiding selection of an appropriate research method. Paper presented at the *Proceedings of the 8th European Conference on Information Systems (ECIS 2000*.

Jenkins, K. (2006). Ignore Islamic Studies at Our Peril. *Diverse Issues in Higher Education, 23*(7), 41.

Jones, J. C. (1992). *Design Methods* (2nd ed.). New York, USA: John Wiley & Sons, INC. (Original work published 1970).

Kahera, A., Abdulmalik, L., & Anz, C. (2009). *Design Criteria for Mosques and Islamic Centers Art, Architecture and Worship*: Elsevier Architectural Press Imprint.

Kalay, Y. E. (2004). *Architecture's new media: Principles, theories, and methods of computer-aided design*. Cambridge, Mass: MIT Press.

Kalay, Y. E. (2006). The impact of information technology on design methods, products and practices. *Design Studies, 27*(3), 357–380.

Kaplan, C. (2000). Computer-generated Islamic star patterns, *In: Bridges 2000, Mathematical Connections in Art, Music and Science, UW School of Computer Science*.

Kaplan, C. S. (2002). *Computer graphics and geometric ornamental design.* Unpublished PhD Thesis, University of Washington, Washington.

Kerr, R. M. (1865). *On ancient lights, and the evidence of surveyors thereon: With tables for the measurement of obstructions*. London: J. Murray.

Kitano, H. (1990). Designing Neural Networks Using Genetic Algorithms with Graph Generation System. *Complex Systems Journal, 4*, 461–476.

Kolarevic, B. (2003). *Architecture in the digital age: Design and manufacturing*. New York ; London: Spon Press.

Kolarevic, B., & Malkawi, A. (2005). *Performative architecture: Beyond instrumentality*. New York: Spon Press.

Kotnik, T. (2006). *Algorithmic Extension of Architecture.* Unpublished Master Thesis, Swiss Federal Institute of Technology, Zurich. Retrieved from http://wiki.arch.ethz.ch/twiki/pub/MAS0506stu/NDSToniKotnik/ToniKotnik_ThesisMAS2006_Small.pdf

Kotnik, T. (2010). Digital Architectural Design as Exploration of Computable Functions. *International Journal of Architectural Computing, 8*(1), 1–16.

Koza, J. R. (1994). *Genetic programming II: Automatic discovery of reusable programs*. Cambridge, Mass: MIT Press.

Kuroishi, I. (2009). Mathematics for/from Society: The Role of the Module in Modernizing Japanese Architectural Production. *Nexus Network Journal, 11*(2), 201–216.

Langley, P., Simon, H. A., Bradshaw, G. L., & Zytkow, J. M. (1987). *Scientific discovery: Computational explorations of the creative process*: MIT Press.

Lawson, B. (1997). *Design in Mind* (2nd ed.). Oxford: Architectural Press. (Original work published 1994).

Lawson, B. (2004). *What Designers Know* (First ed.). Burlington,MA: Elsevier, Architectural Press.

Lawson, B. (2006). *How Designers Think: The Design Process Demystified* (4 ed.). Burlington, MA: Elsevier. (Original work published 1980).

Le Corbusier (2008). *Toward an architecture*. London: Frances Lincoln. (Original work published 1931).

Levin, P. H. (1964). Use of graph to decide the optimum layout of buildings. *Architect Journal, 140*, 809–815.

Liddament, T. (1999). The computationalist paradigm in design research. *Design Studies, 20*(1), 41–56.

Lim, J. (2009). *Bio-structural analogues in architecture*. Amsterdam: BIS Publishers.

Lindenmayer, A. (1968). Mathematical models for cellular interaction in development: Parts I and II. *Journal of Theoretical Biology, 18*, 280-–315.

Liu, H., & Frazer, J. H. (2002). Supporting evolution in a multi-agent cooperative design environment. *Advances in Engineering Software, 33*(6), 319–328.

Liu, Y. T. (2000). Creativity or novelty? Cognitive-computational versus social-cultural. *Design Studies, 21*(3), 261–276.

Madrazo, L. (1994). Durand and the Science of Architecture. *Journal of Architectural Education, 48*(1), 12-24.

March, S. T., & Smith, G. F. (1995). Design and natural science research on information technology. *Decision Support Systems, 15*(4), 251–266.

Marx, J. (2000). A proposal for alternative methods for teaching digital design. *Automation in Construction, 9*(1), 19-35.

Michalewicz, Z. (1996). *Genetic algorithms + data structures = evolution programs* (3rd rev. and extended ed.). Berlin New York: Springer Verlag.

Mokhtar, A., & Ahmadi, A. A. (2005). *Design guidelines for ablution spaces in mosques and Islamic praying facilities*. Sharjah, United Arab Emirates: American University of Sharjah.

Newell, A., Shaw, J. C., & Simon, H. A. (1958). The process of creative thinking. Paper presented at the *Symposium on creative thinking, University of Colorado*. Boulder, Colorado: The Rand Corparation.

Norgaard, R. B. (1989). The case for methodological pluralism. *Ecological Economics, 1*(1), 37–57.

Nunamaker Jr, J. F., & Chen, M. (1990). Systems Development in Information Systems Research *System Sciences, 1990, Proceedings of the 23rd Annual Hawaii International Conference on System Sciences* (pp. 631–640). IEEE Computer Society Press.

Nunamaker Jr, J. F., Chen, M., & Purdin, T. D. M. (1990). Systems Development in Information Systems Research [Article]. *Journal of Management Information Systems, 7*(3), 89–106. from bsh database.

Oxman, R. (2006). Theory and design in the first digital age. *Design Studies, 27*(3), 229–265.

Oxman, R. (2008). Digital architecture as a challenge for design pedagogy: Theory, knowledge, models and medium. *Design Studies, 29*(2), 99–120.

Palladio, A. (1997). *The Four Books on Architecture* (R. Tavernor & R. Schofield, Trans.). Cambridge, Mass: MIT Press. (Original work published 1570).

Peter, S. D., & Ian, F. (2007). Was Waldram wrong? *Structural Survey, 25*(2), 98.

Polkinghorne, D. (1983). *Methodology for the human sciences : Systems of inquiry*. Albany: State University of New York Press.

Rasheed, K. M. (1998). *GADO: A Genetic Algorithm For Continuous Design Optimization.* Unpublished Ph.D dissertation, The State University of New Jersey, New Brunswick, New Jersey.

Rittel, H. (1968). Theories of Cell Configurations: Comments on the Paper by Grason. Paper presented at the *The Design Methods Group First International Conference*. Cambridge, Massachusetts: MIT Press.

Rittel, H. W. J. (1972). *On the planning crisis: Systems analysis of the 'first and second generations'*. Berkeley, California: Univ. of California. (Original work published Reprinted from BedriftsØkonomen, no.8, 1972 (Norway) pp. 390-396).

Rittel, H. W. J., & Webber, M. M. (1973). Dilemmas in a general theory of planning. *Policy Sciences, 4*(2), 155–169.

Robbins, E., & El-Khoury, R. (2004). *Shaping the city : Studies in history, theory and urban design*. New York, NY.: Routledge.

Rosenman, M. A. (1997). The generation of form using an evolutionary approach. Paper presented at the *In Evolutionary Algorithms in Engineering Applications*: SpringerVerlag.

Rosenman, M. A., & Gero, J. S. (1985). Design codes as expert systems. *Computer-Aided Design, 17*(9), 399–409.

Rowe, P. G. (1998). *Design thinking*. Cambridge, Mass.: MIT Press. (Original work published 1987).

Ruch, J. (1978). Interactive Space Layout: A Graph Theoretical Approach. *15th Conference on Design Automation* (pp. 152–157). Las Vegas: IEEE Press.

Rush, R. D., & American Institute of Architects (1986). *The Building systems integration handbook*. New York: Wiley.

Saheeh International (1997). *The Qur'an: Arabic text with corresponding English meanings = al-Quran al-karim ma`a tarjamat al-ma`ani bi-al-lughah al-Injiliziyah*. Jeddah, Saudi Arabia: Abul-Qasim Pub. House.

Schön, D. A. (1988). Designing: Rules, types and words. *Design Studies, 9*(3), 181–190.

Shah, J. J., & Mäntylä, M. (1995). *Parametric and feature based CAD CAM: concepts, techniques, and applications*. New York Wiley.

Shaikh, A., & Kang, S. (1997). Destination driven routing for low cost multicast. *Selected Areas in Communications, IEEE Journal on, 15*(3), 373-381.

Shea, K., Aish, R., & Gourtovaia, M. (2005). Towards integrated performance-driven generative design tools. *Automation in Construction, 14*(2), 253–264.

Simon, H. A. (1983). Discovery, invention, and development: Human creative thinking. *Proceedings of the National Academy of Sciences of the United States of America, 80*(14), 4569–4571.

Simon, H. A. (1988). Creativity and motivation: A response to Csikszentmihalyi. *New Ideas in Psychology, 6*(2), 177–181.

Simon, H. A. (1996). *The sciences of the artificial* (3rd ed.). Cambridge, Mass: MIT Press. (Original work published 1969).

Souza, C. B. d., & Knight, I. (2007). Thermal Performance Simulation from an Architectural Design Viewpoint *Building Simulation Conference* (pp. 87–94). Beijing, China: International Building Performance Simulation Association (IBPSA). from http://www.ibpsa.org/m_bs2007.asp

Stahl, F. I., Wright, R. N., Fenves, S. J., & R. Harris, J. (1983). Expressing standards for computer aided building design. *Computer-Aided Design, 15*(6), 329–334.

Steadman, P. (1983). *Architectural morphology : an introduction to the geometry of building plans*. London: Pion.

Sternberg, R. J. (1999). *Handbook of creativity*. Cambridge, U.K: Cambridge University Press.

Sun, J. (2002). *A framework for supporting generative product design using genetic algorithms.* Unpublished PhD, Hong Kong Polytechnic University, Hong Kong. Retrieved from ProQuest Dissertations & Theses (PQDT) database.

Tang, A., & van Vliet, H. (2009). Modeling constraints improves software architecture design reasoning. *European Conference on Software Architecture. WICSA/ECSA 2009. Joint Working IEEE/IFIP Conference* (pp. 253–256).

Tang, M., & Yao, X. (2007). A Memetic Algorithm for VLSI Floorplanning. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on, 37*(1), 62–69.

Tang, M. X., & Frazer, J. (2001). A representation of context for computer supported collaborative design. *Automation in Construction, 10*(6), 715-729.

Tanghe, J., Vlaeminck, S., Berghoef, J., & Southam, R. (1984). *Living cities: a case for urbanism and guidelines for re-urbanization*. New York: Pergamon.

Taylor, I. A. (1959). The nature of the creative process. In P. Smith (Ed.), *Creativity: an examination of the creative process: a report on the third Communications Conference of the Art Directors Club of New York* (pp. 51–82). New York: Hastings House. from UQ.

Taylor, I. A. (2009). A Retrospective View of Creativity Investigation. In J. W. Getzels & I. A. Taylor (Eds.), *Perspectives in creativity* (pp. 1-36). New Brunswick, NJ, USA: Aldine Transaction. from google books.

Teague Jr, L. C. (1968). Network Models of Configurations of Rectangular Parallelepipeds. Paper presented at the *The Design Methods Group First International Conference*. Cambridge, Massachusetts: MIT Press.

Ternoey, S., & Solar Energy Research Institute (1985). *The Design of energy responsive commercial buildings*. New York: Wiley.

Terzidis, K. (2006). *Algorithmic architecture*. Oxford, UK: Architectural Press.

The University of Illinois, & The Ernest Orlando Lawrence Berkeley National Laboratory (2010). Getting Started with EnergyPlus. In The United States Department of Energy (Ed.) (pp. iv–76): Energy Efficiency and Renewable Energy (EERE) Information Center.

Torrance, E. P. (1995). *Why fly?* Norwood, NJ: Ablex Pub. Co.

Value Engineering Section (1989). Value Engineering report: GDMW Standard Mosques: Various Locations. Riyadh, Saudi Arabia: General Directorate of Military Works (GDMW),.

Wallas, G. (1949). *The art of thought* (3rd. ed.). London, England: Watts & Co. (Original work published 1926 London: Cape Ltd.).

Watson, D., Crosbie, M. J., & Callender, J. H. (1997). *Time saver standards for architectural design data*. New York: McGraw Hill.

Wertheimer, M. (1959). *Productive thinking* (Enlarged ed.). Westport, Conn.: Greenwood Press.

Wong, S. S. Y., & Chan, K. C. C. (2009). EvoArch: An evolutionary algorithm for architectural layout design. *Computer-Aided Design, 41*(9), 649–667.

Wu, S., Lee, A., Koh, W. W. I., Aouad, G., & Fu, C. (2004). An IFC-based space analysis for building accessibility layout for all users. *Construction Innovation, 4*(3), 129–141.

Yang, F., & Bouchlaghem, D. (2010). Genetic Algorithm Based Multiobjective Optimization for Building Design. *Architectural Engineering and Design Management, 6*(1), 68.

Yang, L. Z., Zhao, D. L., Li, J., & Fang, T. Y. (2005). Simulation of the kin behavior in building occupant evacuation based on Cellular Automaton [doi: DOI: 10.1016/j.buildenv.2004.08.005]. *Building and Environment, 40*(3), 411–415.

Yang, R. J., Gu, L., Tho, C. H., Choi, K. K., & Youn, B. D. (2002). Reliability Based Multidisciplinary Design Optimization of Vehicle Structures Paper presented at the *International Conference on Statistics and Analytical Methods in Automotive Engineering*. IMechE HQ, London, UK Bury St Edmunds: Professional Engineering Pub. for IMechE.

Yin, S., & Cagan, J. (2000). An Extended Pattern Search Algorithm for Three-Dimensional Component Layout. *Journal of Mechanical Design, 122*(1), 102-108.

Zacharias, J. (2001). Pedestrian Behavior Pedestrian Behavior and Perception in Urban Walking Environments. *Journal of Planning Literature, 16*(1), 3–18.

Zarrabi-Zadeh, H. (2007–2010). *eQibla* http://eQibla.com