GENERATIVE DESIGN FOR BIM

Its Influence in the Design Process

Naim Korqa

Thesis to obtain the Master of Science Degree in

Architecture

Supervisor: Prof. Dr. António Paulo Teles de Menezes Correia Leitão
Supervisor: Prof. Dr. Francisco Manuel Caldeira Pinto Teixeira Bastos

Examination Committee
Chairperson: Prof. Dr. Teresa Frederica Tojal de Valsassina Heitor
Supervisor: Prof. Dr. Francisco Manuel Caldeira Pinto Teixeira Bastos
Members of the Committee: Prof. Dr. Vitor Manuel de Matos Carvalho Araüjo

November 2015
I would like to express my gratitude:

To Professor António Paulo Teles de Menezes Correia Leitão, for his orientation, dedication, patience and for everything he taught me during the development of this work.

To Professor Francisco Manuel Caldeira Pinto Teixeira Bastos, for the availability, support, dedication, and orientation on the development of this work.

To the members of GAC, for their comments and advices.

To my friends, for their understanding and patience.

To my family, for their support and concern during the development of this work.

To my wife, for her constant support and comprehension.
Architecture is a process of continuous change and innovation, which is influenced by technology. The traditional paper-based design has been replaced by Computer Aided Design (CAD) at the end of XX century, facilitating the editing and revision over the process. Nevertheless, conventional CAD tools do not support change efficiently, limiting the exploration of complex solutions.

The introduction of the Building Information Modelling (BIM) approach in the beginning of the XXI century, was a revolution in the design process, due to its capacity to handle change and mainly due to production and integration of the whole design information. Still, it only worked well up to a certain level of project complexity.

In addition, Generative Design (GD) was introduced in the design process as a disruptive approach that efficiently handles change and data complexity. Nevertheless, it requires significant changes in the traditional workflow of the architect. It is the intent of this dissertation to study these changes in the design process and also propose new design methods that take advantage of the GD approach.

This thesis analyses the current design processes and also the impact of new technologies, with its main perspective towards BIM approach. By taking advantage of BIM, we propose new computational methods for design and a new workflow to achieve better performance in architecture.

Keywords: Generative Design, Building Information Modeling, Computational Methods, Computer Aided Design
Arquitetura é um processo de contínua inovação e mudança, o qual é influenciado pela tecnologia. O desenho tradicional em papel foi substituído pelo Desenho Assistido por Computador (CAD) no final do século XX, o qual facilitou alterações e revisões ao longo do processo de desenho. Ainda assim, as ferramentas tradicionais CAD, uma vez que são ferramentas unicamente de desenho, não lidam de forma eficiente com a mudança, limitando deste modo a exploração de soluções alternativas por parte dos arquitetos.

A introdução da abordagem Modelação de Informação de Construção (BIM) no início do século XXI, foi uma revolução no processo de design, devido à sua melhor capacidade para lidar com a mudança no desenho e, principalmente, pela produção e integração de toda a informação do projeto no desenho. No entanto, as ferramentas BIM funcionam bem só até um certo nível de complexidade.

O Desenho Generativo (GD) assente na escrita de programação que afecta directamente o modelo projectado e assume-se como novo paradigma na arquitetura que, para além de aceitar facilmente a mudança, possibilita a produção de desenhos complexos, os quais seriam quase impossíveis de produzir usando as ferramentas tradicionais CAD. Tem o potencial para a produção eficiente de construções sofisticadas, contudo, o GD pressupõe mudanças significativas no método de trabalho do arquiteto. Com esta dissertação pretende-se estudar essas mudanças no processo de desenho tradicional, como também propor novos métodos para melhor tirar partido da GD.

A tese estuda os processos atuais de desenvolvimento de projeto e do impacto que estas novas tecnologias estão a trazer, com ênfase nas tecnologias BIM. Portanto, tirando partido das ferramentas BIM, vamos propor novos métodos computacionais no processo design, para alcançar melhor desempenho na arquitetura.

Palavras-chave: Desenho Generativo, Modelação de Informação de Construção, Métodos Computacionais, Desenho Assistido por Computador.
# TABLE OF CONTENTS

ACKNOWLEDGMENTS ................................................................. iii
ABSTRACT ...................................................................................... v
RESUMO ......................................................................................... vii
TABLE OF CONTENTS ................................................................. ix
LIST OF FIGURES ........................................................................... xii
LIST OF TABLES ............................................................................. xv
ABBREVIATIONS ........................................................................... xvii
GLOSSARY OF TERMS ................................................................. xvii

## INTRODUCTION ........................................................................... 1

OBJECTIVES ................................................................................... 4
METHODOLOGY ................................................................................ 4
STRUCTURE ......................................................................................... 5

## PART-I: BACKGROUND .............................................................. 7

1 COMPUTER AIDED DESIGN ....................................................... 8
   1.1 INTRODUCTION ....................................................................... 8
   1.2 HISTORY OF CAD ................................................................. 8
   1.3 CONCLUSION .......................................................................... 11

2 CAD + SCRIPTING ................................................................. 13
   2.1 INTRODUCTION ....................................................................... 13
   2.2 SCRIPTING TOOLS ............................................................... 13
      2.2.1 AutoLISP ........................................................................ 13
      2.2.2 Rhinoscript ....................................................................... 14
5.4 CONCLUSION.............................................................................................................. 49

6 NEW WORKFLOW IN THE DESIGN PROCESS ......................................................... 50
  6.1 INTRODUCTION........................................................................................................ 50
  6.2 THE MOST SUITABLE USE OF GD METHODS ...................................................... 51
  6.3 NEW WORKFLOW IN THE DESIGN PROCESS ...................................................... 55
  6.4 GENERATIVE DESIGN LIMITS .............................................................................. 58
  6.5 CONCLUSION.......................................................................................................... 59

7 EVALUATION ............................................................................................................ 61
  7.1 INTRODUCTION........................................................................................................ 61
  7.2 LOUVRE ABU DHABI .......................................................................................... 61
  7.3 ABSOLUTE TOWER ............................................................................................. 64
  7.4 CONCLUSION.......................................................................................................... 71

8 CONCLUSIONS AND FUTURE WORKS ................................................................. 73
  8.1 CONCLUSIONS........................................................................................................ 73
  8.2 FUTURE WORKS .................................................................................................... 75

BIBLIOGRAPHY........................................................................................................... 77
Figure 1. Paper-based drawing techniques (source: www.egsolutions.com) ........................................... 1
Figure 2. Project developed in AutoCAD - a CAD software. (source: www.bibliocad.com) ...................... 1
Figure 3. Project in Revit Architecture, a Building Information Modeling tool (source: www.buildipedia.com) .................................................................................................................. 2
Figure 4. Project based on Generative Design methods (source: www.fwallpapers.com) ......................... 3
Figure 5. Monochrome graphic terminals (source: www.design.osu.edu) .............................................. 8
Figure 6. MIT Sketchpad program, ca. 1965 (source: www.computerhistory.org) ................................. 9
Figure 7. The user interface of the first version of CATIA (source: www.floopydisc.tumblr.com) ............ 9
Figure 8. A snapshot of AutoCAD 2.8 (source: Chatzitsakyris, 2015) ..................................................... 10
Figure 9. Keith Bentley begins selling MicroStation (source: www.cadalyst.com) ............................... 10
Figure 10. The user interface of the first Pro/Engineer version (source: Chatzitsakyris, 2015) ............. 10
Figure 11. CATIA version 5 (source: www.fi-video.szlm.com) ................................................................. 11
Figure 12. Autolisp scripting example (source: Celani, 2008) ............................................................... 14
Figure 13. Academic Report- Turning Torso (source: Fantainha, Santos) .......................................... 14
Figure 14. Hexgrid with Rhinoscript (source:http://www.rhinoscript.org/) ...................................... 15
Figure 15. Lou Ruvo Center for Brain Health (source: www.francisandfrancis.com) ....................... 15
Figure 16. Grasshopper and logic element connection (source: www.danieldavis.com) ................... 17
Figure 17. London Design Festival, SCIN Gallery (source: www.matsysdesign.com) .................... 17
Figure 18. A BIM model (source: www.buildipedia.com) ................................................................. 19
Figure 19. ArchiCAD environment (source: www.aecmag.com) .......................................................... 21
Figure 20. Revit Architecture environment (source: www.magicad.com) ........................................... 22
Figure 21. Foundation Louis Vuitton (source: www.letudiantautonome.fr/fondation-louis-vuitton/) .... 23
Figure 22. 3D Model of the Foundation Louis Vuitton (source: www.network.aia.org) ................... 23
Figure 23. BIM master model (source: www.network.aia.org) .............................................................. 24
Figure 24. Interoperability with other software (source: www.network.aia.org) .................................. 24
Figure 25. Project on 8 Spruce Street (source: www.urbanfile.org) ..................................................... 25
Figure 26. The analysis model of the curtain wall (source: Goldklang, 2013) ..................................... 26
Figure 27. Beekman Tower facade panels (source: www.arch-re-review.blogspot.ch) .................... 26
Figure 28. Three ways of applying the generative design system (Tomasz Estkowski) ....................... 29
Figure 29. Graphs representing the sets of configurations generated in the initial steps of evolution according to a typical class 3 cellular automation rule (source: Wolfram, 1984) ...................... 30
Figure 30. Three-dimensionally mapped CA system variations (source: Knight, 2000) ...................... 30
Figure 31. The tree generation through L-system (source: www.cgjennings.ca) .................................. 31
Figure 32. L-System using three modules as leaves (source: www.michael-hansmeyer.com) .......... 31
Figure 33. A planar Voronoi Diagram(solid lines) and the nearest connected point(dashed lines). (source:www.war-worlds.com) .............................................................. 32
Figure 34. The formation of cell spaces by Voronoi System (source: Bahraminejad & Babaki, 2014)...

Figure 35. Ice-ray Grammar (source: Knight, 2000) ................................................................. 32

Figure 36. Hypothetical proposal of the ancient Pompeii, based on real building analysis, through shape grammar system (source: CityEngine, 2015) ................................................................. 33

Figure 37. Mandelbrot Set. (Source: www.math.utah.edu) ........................................................... 34

Figure 38. Five alternatives of shades with different transformations of fractals (source: Sedrez, Meneghel and Celani, 2014) ........................................................................................................... 34

Figure 39. Crossover, in one of the fundamental mechanism of genetic algorithms (source: www.geos.ed.ac.uk) ............................................................................................................................................ 35

Figure 40. Examples of GA facade exploration (source: Gagne & Andersen, 2010)........................... 35

Figure 41. Design Script Editor (source: www.through-the-interface.typepad.com).......................... 39

Figure 42. Dynamo demo’s using a Stadium dataset (source: www.architosh.com) .......................... 40

Figure 43. The furniture defines if the door is locked (source: www.case-inc.com/blog/revit-dynamo-more-than-grasshopper) .................................................................................................................. 41

Figure 44. ETFE facade of Beijing National Aquatic Center (source: www.stylepark.com) ............... 42

Figure 45. Sustainability issues (source: www.mumagi.net) ............................................................ 42

Figure 46. Data transfers at the competition stage (source: BIM Handbook 1st) ............................... 43

Figure 47. Data transfers at the design development stage (source: BIM Handbook 1st) ..................... 44

Figure 48. The Strand7 Finite Element model of the Water Cube. (source: www.aecmag.com) ......... 45

Figure 49. Aviva Stadium (source: www.info-stades.fr) ................................................................. 47

Figure 50. Series of daylight studies to evaluate different building forms and footprint. (source: Eastman, et al., 2011) ......................................................................................................................... 47

Figure 51. Water Cube facade structure variation (source: www.architizer.com) ............................ 51

Figure 52. Beekman Tower plan (source: www.wirednewyork.com) ................................................ 51

Figure 53. Simple parametric plan in Revit (source: help.autodesk.com) ......................................... 54

Figure 54. Optimization of a table by Programming Architecture (source: https://www.youtube.com/watch?v=fws3pKzDpmc) .......................................................................................................................... 54

Figure 55. Macro that calculates heat gain of different types of windows with and without shading in Revit Architecture. (source: www.aecbytes.com) ....................................................................................... 55

Figure 56. Workflow of Traditional and Generative Design process. .............................................. 57

Figure 57. The Louvre Abu Dhabi : a major technical and architectural project by Jean Nouvel, based on BIM and GD methods (source: www.openbuildings.com) ............................................... 61

Figure 58. Interior connection spaces of Louvre Abu Dhabi (source: www.e-architect.co.uk) .............. 62

Figure 59. Overall translucency map with resulting shadow effect beneath the dome. (source: Imbert, et al., 2012) ................................................................................................................................. 62

Figure 60. Dome Structure (source: Imbert, et al., 2012) ............................................................... 63

Figure 61. Pattern of the Dome (source: www.tales-magazine.fr) .................................................... 63

Figure 62. Diagram of parametric workflow (source: Imbert, et al., 2012) ....................................... 64
Figure 63. Absolute Tower exterior image and Plan (source: www.urbantoronto.ca) ..................................65
Figure 64. Absolute Tower model and plan (Revit Architecture/ Programming approach) ..................66
Figure 65. Absolute Tower model with rectangle shape floors (Revit Architecture/ Programming approach) ..................................................................................................................67
Figure 66. Absolute Tower model with octagonal shape floors (Revit Architecture/ Programming approach) ..........................................................................................................................68
Figure 67. Absolute Tower model with more floors (80 levels) (Revit Architecture/ Programming approach) ..................................................................................................................................69
Figure 68. Absolute Tower model with increasing size of floors (Revit Architecture/ Programming approach) ........................................................................................................................................70
LIST OF TABLES

Table 1. Proposal of the most suitable use of GD approach and BIM, .................................................. 52
Table 2. Results of the time needed to create the initial model in BIM and GD approach ..................... 66
Table 3. Results of the time needed to change the floor into a square shape ...................................... 67
Table 4. Results of the time needed to change the floor into an octagonal shape .............................. 68
Table 5. Results of the time needed to change the number of the floors ........................................... 69
Table 6. Results of the time needed to change the size of floors in proportion with the level .............. 70
ABBREVIATIONS

AEC – Architecture, Engineering, and Construction

BIM – Building Information Model

CAD – Computer Aided Design

CAM – Computer Aided Manufacturing

CADAM – Computer Augmented Design and Manufacturing

CADDS – Computer Aided Design and Drafting Software

CNC – Computer Numerical Control

GD – Generative Design

GA – Genetic Algorithm

IGDS – Interactive Graphics Design Software

PDM – Product Data Management

VBA – Visual Basic for Applications

GLOSSARY OF TERMS

Generative Design - The generative design approach can be defined as a process based on rules or algorithms, through which various elements and design possibilities can be created.

Generative System - A system that generate options for design problems.

Generative BIM design - The use of algorithmic approach within the BIM environment.

Algorithmic approach - An approach to design that is controllable and can easily handle change. It allow the generation of several different variations of the same design.

Building Information Modeling - An approach that offers the possibility to build virtually, simulating the construction environment with all the information needed for construction.

Scripting - An approach that allows the user to access the underlying structure of existing software and embed new functionality to it.
**Program** - An algorithm written in a way that the computer understands.

**Programming** - The act of translating the algorithms into a program, so it can be performed by the computer.

**Macros** - They are new commands developed by the users and consist of recording a sequence of commands, to call them when necessary.
INTRODUCTION

Designing is a problem solving process that seeks to address a wide range of issues, often ambiguous in nature. Thus, a fully developed understanding of abstract notions and application of a variety of solutions is considered a norm, leading to a process characterized by numerous modifications.

Historically, designing was highly influenced by the drawing techniques of each era. Until the second half of the XX century, precise technical drawings, perspectives, and draw-by-hand techniques were the central points in the design process. Being labor intensive, modifications in design resulted with prolonged working hours.

The 60s marked an advancement in technology that led to a gradual increase in productivity. Computer Aided Design (CAD) emerged as a new computer field to assist in the project documentation. The traditional CAD systems only produced geometric entities in 2D, by creating and manipulating drawings in monochrome graphic terminals. The first CAD tools used a combination of simple geometries, involving straight lines and circles to represent the desired shape, and did not allow the design of more complex geometries (Schoonmaker, 2002).

CAD applications expanded the efficiency of design activities and allowed architects to deliver more precise drawings (Schoonmaker, 2002) that could be more easily altered without the need of manually deleting and redrawing all of their plans. With the introduction of these new tools, the design process was revolutionized, it opened potential outcomes for additional innovative and masterful models.

Figure 1. Paper-based drawing techniques (source: www.egsolutions.com)

Figure 2. Project developed in AutoCAD - a CAD software. (source: www.bibliocad.com)
Unfortunately, errors were liable to happen, and CAD programs did not have the capacity to recognize them due to the inability of interconnecting different inputs in a project. Changing a complex model was still a hardworking and prolonged task. A minor alteration in the project lead to changes in all the architectural design, such as plans, sections, and facades. Furthermore, these drawings and models used pure geometry with no construction data, causing a documentation problem.

Building Information Modeling (BIM) showed up as a new paradigm of design that introduced an answer to these problems. BIM is an advanced three-dimensional representation of a building that has its information, to support the development, construction, fabrication and procurement activities, made possible by the use of parametric BIM objects containing all the necessary information (Eastman, et al., 2008).

Moreover, a BIM model contains data about the life cycle of the building that can be easily extracted to generate documents. BIM also allows a more efficient design process by facilitating the reuse and sharing of information.

However, the BIM approach still presents some disadvantages. It does not support well complex geometries nor complex design modifications, and it is not flexible enough.
Generative Design (GD) appeared as a response to these problems. GD is a design method that produces shapes through algorithms. Specifically, the GD method is a precise, flexible and controllable tool that allows changes more easily. In addition, it also enables the production and optimization of several solutions and alternatives of the project (Fernandes, 2013).

![Figure 4. Project based on Generative Design methods (source: www.fwallpapers.com)](image)

Unfortunately, there are still issues to overcome in GD tools. The GD approach requires programming knowledge, which most architects do not have. Then, each CAD application has its own programming language that makes it even harder for architects to learn multiple coding languages. Furthermore, there are other limitations such as cost of programming alterations and time of model generation by the programming approach, that also limit the use of GD methods.

In this context, this dissertation argues that the GD approach should be introduced in the design process in a way that increases design efficiency. It requires an initial effort, but as we will show, the GD advantages will later reward the initial effort.
OBJECTIVES
The main intent of this dissertation is to investigate the potential of Generative Design as an auxiliary tool integrated within the design process. In particular, the intent is to explore the changes and influences in the design process.

To this end, we will introduce a workflow that shows how architects will deal with GD and BIM, based on experiences and case study analyzes. In addition, we propose some parameters to define when projects should involve GD and when they should not. These parameters are defined based on (1) the data complexity and (2) the project scale.

Furthermore, we make an evaluation of GD and BIM approach, that consists of two parts, an analysis and a simulation.

In the first part, we present a case study of one of the most recent projects made with the BIM and GD approach, illustrating the successful use of GD methods. In the second part, we present a simulation of modeling to compare the efficiency of GD methods with conventional approaches.

Finally, the implementation of this evaluation aims to prove the efficiency of GD methods, particularly when compared to traditional approaches.

METHODOLOGY
The methodology followed in this dissertation has four stages: 1 - Literature review, 2 – Study of the design process influences 3 – Case study analyses, 4 – Conclusions.

The first stage, the literature review, consist of the collected information regarding the design process, design tools, and their evolution over the last decades. Based on the bibliography, we investigate the first capabilities of digital tools, in parallel with computational advances. Moreover, we describe the evolution and the influence of design tools in the design process. Finally, we consult the recent papers and texts that discuss the integration of GD in the design process.

The second stage consist of all the collected information translated into a final result and also a proposal of how GD methods should be integrated into the design process. The aim of these analyses is to define the advantages and limits of GD and BIM, as well as the possible changes in the design workflow.
In the third stage, we produce a brief evaluation of some complex built projects, which best shows the need for sophisticated tools to make them possible.

In the last stage, we make a conclusion of the entire research as well as a proposal for future investigations.

**STRUCTURE**

This dissertation is divided into two parts: **Background** and **Generative BIM tools in the Design Process**.

The first part, **Background**, consists of 4 chapters:

1. **COMPUTER AIDED DESIGN**

This chapter introduces the first digital design tools, their evolution and also their potential. It illustrates the spectrum from the basic tools to the most recently used tools.

2. **CAD + SCRIPTING**

This chapter contains some historical data on the first uses of scripting and other programming algorithms in design. It describes some examples where scripting was used to assist design as well as the most used scripting tools.

3. **BUILDING INFORMATION MODELING**

This chapter presents a brief history of BIM and its importance in the design industry. Moreover, we analyze some case studies to identify the advantages and the limits of this approach.

4. **GENERATIVE DESIGN**

This chapter presents the main idea of Generative Design with its most well-known methods. Furthermore, it also contains some examples of GD usage based on different GD methods.

The second part, **Generative BIM tools in the Design Process**, is composed of three main chapters:
5. BIM WITH GD

This chapter discusses the combination of the two most powerful approaches, GD and BIM. It presents some related case studies, where this combination was irreplaceable, outlining the advantages and limits of the approach.

6. NEW WORKFLOW IN THE DESIGN PROCESS

This part contains an analysis of all the collected data and a proposal for the future workflow of designing, including GD methods. It describes how the designer will use the GD tools and how it will influence the design process.

7. EVALUATION

In this chapter we develop an evaluation of the GD and BIM approach, based on two built projects. In the first project we analyze the use of GD methods. Furthermore, in the second project we make a brief simulation of modeling comparing the efficiency between GD and BIM approach.

8. CONCLUSIONS AND FUTURE WORK

Finally, in the last chapter we present the conclusion of the Thesis and propose some guidelines for future investigations.
PART-I: BACKGROUND
1 COMPUTER AIDED DESIGN

1.1 INTRODUCTION
The use of digital technologies has transformed the design and the production methods. With the introduction of Computer Aided Design (CAD) in the design process, the graphical representation of designs evolved, enabling the realization of the most detailed drawings, with greater efficiency.

Moreover, CAD tools allowed more flexible design reviews, and all the drawing of one stage could be used as base to the development of the next stage. So, it allowed a good integration between design stages. Furthermore, CAD tools allowed to edit or repeat drawings more easily, without having to manually delete and redraw.

Initially, CAD tools offered mere 2D drawings to assist graphical representation that later advanced into 3D modeling, allowing greater exploration of design alternatives.

The next section presents a brief history of the evolution of computational design tools from the beginning of the 1960s until the beginning of the 2000s, highlighting the key programs and events in the CAD development.

1.2 HISTORY OF CAD
The beginning of Computer Aided Design can be traced back to 1950 when the Massachusetts Institute of Technology started the development of digital drawing tools (Encarnação, 1990). CAD systems of this generation were limited to the description of geometric entities in two dimensions with the creation and manipulation of drawings in monochrome graphic terminals (Fig. 5).

Figure 5. Monochrome graphic terminals (source: www.design.osu.edu)
These CAD systems used a combination of simple geometries, involving straight lines and circles to represent the desired shape, which did not allow the design of complex geometries.

In 1957, Dr. Patrick Hanratty developed the programming tool “Pronto”, which was one of the first CAD systems. Thereafter, Ivan Sutherland developed “Sketchpad” in 1960, the first program to be able to interact with a light pen and it was used to design mechanical parts (Cohn, 2010).

The main purpose of the first generation of CAD software was to automate repetitive 2D drawings, so automotive and aerospace companies were the earliest commercial users of CAD due to the high cost of computers (Broquetas, 2010).

CAD software finally migrated into commercial use in the 1970s and one of the most famous 2D CAD software program of this time was CADAM (developed by the Lockheed aircraft company). In 1977 Lockheed developed a 3D CAD software program named CATIA, which is still applicable to this day.

Even though CATIA was the famous 3D modeling software, the first 3D solid modeling program was SynthaVision from MAGI (Mathematics Application Group) in 1972; it was a mere 3D analysis of nuclear radiation exposure (Weisberg, 2008).

The success and growth of CAD software increased the need for standardization. Thereby, IGES (Initial Graphic Exchange Standard) was created in 1979, to facilitate the transfer of 3D curves and surfaces between different CAD software programs.

Initially, IGES was used to exchange 2D/3D wireframe models, dimensioning data, text and limited classes of surfaces, which was considered as an awful experience by many users. Thereafter, it was gradually extended and developed focusing on clarity and consistency, and allowing a better exchange between CAD applications.

The competition between different CAD software companies led to many new CAD applications seeking to capture a share in the new market of CAD software, such as Auto-draft, Calma, CADDS, IBM’s CADAM, M&S Computing’s IGDS, McAuto’s Unigraphics and Auto-Trol (Weisberg, 2008).
The decade of the 1970s saw major advances in CAD software, primarily because of the increasing power of computers. Computer affordability combined with improving software had a great impact on small companies.

Personal Computers(PC) appeared in the early 1980s, with the first one shipped in 1981. Thereafter, Autodesk created the first CAD software for PCs in 1982 - the AutoCAD Release 1 (Cohn, 2010).

Furthermore, in 1984 Bentley System developed and released MicroStation, which was developed based on another CAD package of the time and provided an advanced computer aided design on PC. Apple released the first Macintosh 128 in 1984, and in 1985, MiniCAD was released by Biehl Graphisoft, which rapidly became the most used CAD software in Mac Computers (Cadazz, 2015).

In the 1980s, the CAD software industry made incremental improvements taking advantage of continuing development in computer hardware. A new 3D solid modeling software, Unix 3D CAD Pro/Engineer, launched by Parametric Technology Corporation appeared in the market, and the industry would never be the same again (Cadazz, 2015).

Pro/Engineer came as a revolution in the CAD tools with its interface, facility of use and speed of modeling. Moreover, Pro/Engineer was the first CAD software to be entirely based on solid models, history-based features, and constraints.

While in the past, 3D CAD software took years and millions of dollars to be developed, in 1993 it was possible to be developed with a smaller budget.
and in less than a year. That was proved by the small software company called Solid Works.

Furthermore, the mid-1990s was marked by two significant changes: the PDM (Product Data Management) systems and the 3D CAD software PC explosion (Dias Calado do Amaral, 2010). Autodesk, under pressure to improve the 3D CAD software, released Mechanical Desktop, offering a first full-functioning 3D solid modeling CAD software product that rapidly became the number one 3D CAD software in the world (Cadazz, 2015).

The last marked event of the 1990s in the CAD software was standardization on Unigraphics, which also marked a new phase in CAD development (Cadazz, 2015). The fast growth of CAD tools was over, and the level of differentiation between them was ever more difficult and expensive. In addition, this tough competition became the main reason to reduce the price of CAD software.

By the end of 1990s, 3D modeling across the web brought a new era of CAD applications, Dassault Systems, with CATIA Conferencing Groupware, enabled a better collaboration and review of CATIA models via the internet (Cadazz, 2015).

In 1999, Dassault Systems released its long waited CATIA Version 5, the first software to be fully implemented in Windows. In turn, all the leading companies did the same (Cadazz, 2015).

Thereafter, the Parametric Technology strategically merged with the Computervision to achieve the major share in the CAD software market. Thus, it became present into the biggest AEC industries.

With Parametric Technology, about 37 years after Ivan Sutherland published his SketchPad thesis, CAD software industry entered in a stable period of development (Cadazz, 2015). Even though CAD applications continue to be developed, the improvements were subtle and insignificant.

1.3 CONCLUSION
The constant evolution of CAD tools clearly changed the design practice. In the past, paper-based drawings were extremely laborious to draw, edit or change. Then with the introduction of CAD tools, the drawings could be more
easily done, altered or redraw. Therefore, CAD tools became a more efficient replacement of paper-based technique.

The development of CAD software started by offering simple geometric elements such as straight lines and circles, and later advanced to the 3D modeling, thus, increasing the drawing efficiency and consistency, but without any information regarding the design or construction.

Furthermore, after the CAD development achieved a level of maturity, the improvements began to be very subtle. Therefore, the designers felt the need for additional tools, to accomplish their tasks with more efficiency.

To this end, more advanced design approaches were developed, as will be shown in the following chapters.
2 CAD + SCRIPTING

2.1 INTRODUCTION

Despite the sophistication of CAD tools, they are still a long way from addressing all the designer needs. An answer to this problem was provided by scripting languages, which allow the manipulation of CAD tools.

Scripting languages allows the user to access the underlying structure of existing software and embed new functionality in it (Kashyap, 2004). So, they are usually used to assist the design by simplifying repetitive tasks, by creating new features in the software, by performing tasks more efficiently or also by manipulating large amounts of data.

Scripting languages for CAD software vary widely, not only in terms of their syntax and structure but also on the results obtained from these applications. Some of the currently most used scripting languages for CAD applications are AutoLISP, Rhinoscript, and Grasshopper.

2.2 SCRIPTING TOOLS

2.2.1 AutoLISP

AutoLISP is a programming language designed to adapt AutoCAD functionality, and it was first introduced in the mid of the 1980s, as an application programming interface (API) in AutoCAD Release 2.1 (Autodesk, 2015).

AutoLISP can control every operation of AutoCAD, from the appearance of the workspace to the variety of tools available. It can be very useful to create more tools that increase AutoCAD productivity. In addition, it also allows the exploration of more complex geometries, which the conventional CAD tools are not capable to produce.

However, massive effort is required for utilizing the true potential of this programming language.
Fig. 12. Autolisp scripting example (source: Celani, 2008)

Fig. 13 shows an example produced with AutoLISP and it was developed as an academic work in the Programming for Architecture course of Master in architecture of Instituto Superior Tecnico (IST). The model consisted of a building that twists according to an angle that increases with height. Also, walls dimensions, windows and floors are constrained by rules defined in the program which can be controlled by changing the program's parameters.

Fig. 13. Academic Report- Turning Torso (source: Fantainha, Santos)

As illustrated in this example, AutoLISP is a crucial asset of AutoCAD, since it enhances and extends its capabilities, allowing the user to explore more possibilities and execute tasks efficiently. Nevertheless, this scripting language is not used frequently by designers since it requires programming knowledge, a skill that most of them do not have.

2.2.2 Rhinoscript

Rhinoceros 3D provides a command line interface, which means that its functions can be commanded by keyboard. The simplest form of programming available in Rhinoceros 3D are Macros. They are new
commands developed by the users and consist of recording a sequence of commands, to call them when necessary.

When the user uses Macro, all the sequence of commands that defines it are executed. This feature is easy to use but very limited, since the commands must be written over and over in the macro definition.

These limitations led to the development of scripting languages, which are something between macros, programs, and plug-ins. One of the most popular scripting languages available in Rhinoceros 3D is Rhinoscript due to its high level of features (Rutten, 2007).

Rhinoccript is a scripting tool that allows access to Rhino’s geometric library, procedures, and graphical user interface through the Visual Basic scripting language, considered by programming users as relatively easy to use (Rutten, 2007).

The Lou Ruvo Center for Brain Health, a building designed by Frank Gehry, is an example of a successful project that took advantage of Rhinoscript. The 18,000 stainless steel shingles, each cut to unique measurements and 199 different windows, make this project stand out for its complexity.

The possibility to access the whole library of Rhinoceros 3D makes Rhinoscript one of the most used scripting languages, especially when it comes to complicated shapes such as the above project. Therefore, it is related exclusively to the modeling of complex shapes and do not produce any other information related to the design process.
In addition, using Rhinoscript is also possible to work collaboratively because the code can be structured with functions, which are defined separately and can be retrieved at any time (Shearer, 2010).

Nevertheless, Rhinoscript requires programming knowledge, which is an obstacle to many designers. Therefore, a solution to this lack of programming knowledge was the possibility to use visual programming, as explained in the next section.

2.2.3 Grasshopper

Grasshopper is a visual programming language developed by architects as a plug-in for Rhinoceros 3D. The origins of Grasshopper can be traced back to the “Record History” button, available in the fourth version of Rhinoceros 3D. This feature saved the modeling procedure steps in the background. Then in 2008, David Rutten posed the question “what if we can have more explicit control over the modeling procedure history”. To answer this question, Explicit History, the precursor of Grasshopper, was developed (Akos & Parsons, 2014).

After that, Grasshopper was developed and it became a graphical editor integrated in Rhinoceros 3D. This tool does not require programming skills, since it allows users to structure their program with components that represent parameter or forms. These components are connected among themselves by lines that also establish a dataflow. The program is executed following this dataflow and the results are produced in Rhinoceros 3D (Akos & Parsons, 2014).

In addition, Grasshopper became very popular amongst architects because of its graphical interface and how easy it is to learn. However, this visual language has a significant disadvantage. As the model grows in complexity, the number of connections between components makes it difficult to understand (Fig.16). If there is a need to change something, it becomes almost impossible to edit the model.
An example created with Grasshopper is the built project developed by SCIN, a material resource center for designers and architects in London. The objective was to produce a small cube (Fig. 17) that represents their approach to design, materiality, and technology for the London Design Festival exhibition. The cube was designed using a network of digital cellular bodies that are first relaxed to produce a more uniform field and then structurally differentiated about their distance to the outside surface. The inner core’s cell edges are extremely thin and fragile, yet are protected by the multiple layers of more robust edges closer to the cube boundary (Matsysdesign, 2015).
Grasshopper was a revolution in generative design and continues to be one of the most used tools in the market. Still, this tool is only useful in the beginning of the project because it does not work well in more advanced phases of the design process.

Finally, it is also very difficult to work in teams using Grasshopper, because it is almost impossible to divide the model into parts, and models are difficult to comprehend, specially the highly complex ones.

2.3 CONCLUSION

It is clear that scripting languages are developed in search of improving the user experience with software, however, the technological advancements have yet to guarantee meeting fully the designer needs.

The first two examples of scripting languages, AutoLisp and Rhinoscript, are textual languages for AutoCAD and Rhinoceros 3D. Those scripting tools can access the underlying structure of existing CAD software and embed new functionality in it. Yet, for this to work the user must have knowledge about scripting. Therefore, since most designers do not know scripting, it becomes a barrier for them to use these tools.

The third example, Grasshopper, is a visual programming language for Rhinoceros 3D. It was created specifically to assist the designers that do not know textual programming. As a result, it is currently one of the most used scripting tools. Yet, it has a significant disadvantage. It works well just in the initial phases, because as the program grows, it becomes impossible to understand it or to make changes on it.

Furthermore, another disadvantage of these scripting tools is that since they were created to support CAD applications, they are usually used to create geometry, without any other construction or design information.

An answer to these problems was provided by the Building Information Modelling, as will show in the following chapter.
3 BUILDING INFORMATION MODELING

3.1 INTRODUCTION

Design is constantly improving its methods and techniques, aiming to increase quality and efficiency. Thereby, the amount of data also increased and became a new challenge for the AEC industry.

A promising idea that has potential to achieve this goal is Building Information Modeling (BIM). The BIM approach offers the possibility to build virtually, simulating the construction environment with all the information needed for construction (Azhar, 2011).

Unlike CAD, instead of producing simples 3D geometry, BIM produces parametric objects containing all the necessary information for construction. This information can be extracted from the BIM model at any time during the design process, including plans, materials, dimensions, cost estimates, project schedules and analyses. Moreover, this information serves to support all the involved parties in the design and construction processes, from contractors, to builders, to clients, and even to the maintenance staff of post-construction.

Another advantage of BIM is the accuracy and consistency of the design. Since we work with a 3D parametric model and all the views are interconnected, any alteration in one will be automatically updated in all the other views. Furthermore, it has the potential to virtually check the 3D model for clashes. This feature reduced significantly the waste and errors in the construction, by identifying them in advance.

The next section presents a brief history of the evolution of BIM tools from the beginning of the 1980s until the beginning of the 2000s, highlighting the key programs and events in the BIM approach development.

3.2 HISTORY OF BIM

BIM is perceived as a new paradigm of designing, modeling, and building, but this concept is not recent at all. In the earliest years of computing, in 1962, Douglas C. Engelbart published the paper *Augmenting Human Intellect*, where he described his vision of the future of architecture and presented some of his ideas and concepts, such as:
the architect next begins to enter a series of specifications and data—a six-inch slab floor, twelve-inch concrete walls eight feet high within the excavation, and so on. When he has finished, the revised scene appears on the screen. A structure is taking shape. He examines it, adjusts it... These lists grow into an evermore-detailed, interlinked structure, which represents the maturing thought behind the actual design"

In 1975, a concept very similar to the BIM approach was first documented by a working prototype called “Building Description System”. It was published in the AIA Journal by Charles M. Eastman and several concepts of BIM were also mentioned,

“. . . interactively defining elements . . . deriv[ing] sections, plans, isometrics or perspectives from the same description of elements . . . Any change of arrangement would have to be made only once for all future drawings to be updated. All drawings derived from the same arrangement of elements would automatically be consistent . . . any type of quantitative analysis could be coupled directly to the description . . . cost estimating or material quantities could be easily generated . . . providing a single integrated database for visual and quantitative analyses . . . automated building code checking in city hall or the architect’s office. Contractors of large projects may find this representation advantageous for scheduling and materials ordering.”

Meanwhile, in the early 1980s, other parallel researches about BIM were conducted in Europe and in the USA. While in Europe, this concept was named as “Product Information Models”, in the USA it was described as “Building Product Models”. Thereon, these two nomenclatures later merged together into “Building Information Model” (Eastman, et al., 2008).

The first use of the term “Building Modeling”, in the sense of BIM was in a paper by Robert Aish in 1986. This document describes a case study to which the Building Modeling System was applied, illustrating the arguments and concepts of BIM approach that we know today. From there, it was a short leap into the term “Building Information Model”, which was first introduced in the paper “Modeling Multiple Views on Buildings” by G.A. van Nederveen and F. Tolman, in 1992 (Eastman, et al., 2008).

In parallel with the development of the concept and nomenclature, several software applications (Brics, US-based Bausch & Lomb modeling system,
Rucaps) implemented these changes into their functionalities, although most of them are not remembered today (Eastman, et al., 2008).

The first true BIM software, Radar CH, which later became ArchiCAD was developed by Gabor Bojar in Hungary. Even though ArchiCAD was the world’s first BIM software, it did not become very popular until recent years, due to computing limitations of the time as well as the unfavorable market trends (Quirk, 2012).

Figure 19. ArchiCAD environment (source: www.aecmag.com)

After ArchiCAD release, the Parametric Technology Corporation (PTC) was established and produced their first constraint-based parametric software PRO/ENGINEER.

Thereafter, two co-workers of PTC, Irwin Jungreis, and Leonid Raiz, whom already owned the know-how of Pro/Engineer, split from PTC and established their own software company. Their goal was to create software that could deal with bigger and more complex projects than ArchiCAD could.

So, in 2000 they developed a software called Revit, which utilized a parametric change engine. In 2002, the company was sold to Autodesk, which resulted in Revit being more aggressively promoted and more sophisticated (Quirk, 2012).

Revit brought many innovations that revolutionized the market, such as parametric families, construction phase control, schedules and visual programming environment. Moreover, Autodesk invested in collaborative design, to improve the collaboration between large teams of architects, engineers, and contractors. In addition, by 2004, with Revit 6 they released
Revit Structure and Revit MEP to improve the communication between engineers and architects (Quirk, 2012).

Since there are different BIM software in the market, it became difficult to exchange data from different BIM platforms. Therefore, to combat this problem, in 1995 the International Foundation Class (IFC) was developed. IFC has still some exchange limitation, but it is constantly adapting and improving model exchange.

After some years of BIM development, in 2005, the first industry-academic Conference on BIM has been held, (Eastman, et al., 2008) where a broad range of software designers and vendors, as well as successful BIM tools users, were present, showing their achievements.

To better understand how BIM can help us design and build complex projects, the following section of the thesis will describe two case studies where the use of BIM tools was essential to their success. The two selected projects are the Foundation Louis Vuitton in Paris and the Beekman Tower in Manhattan.

3.3 BUILT CASE STUDY OF BIM
3.3.1 Foundation Louis Vuitton
The new France Art Museum, Foundation Louis Vuitton, designed by Frank Gehry, is located on the edge of the Jardin d’Acclimatation in Paris and it is considered a masterpiece of design, art and technology.
This building is composed of several exhibition galleries of different shapes, as well as a complex modular auditorium, allowing different configuration use. The main hall acts as a guiding point, connecting the building’s different spaces, such as the galleries, the auditorium, the library, the restaurant and the terraces.

An important aspect of this project was the use of glass, which not only has a strong presence in Paris and provides a rich interaction with nature, but is also a reference to historical garden structures of the 19th century (Fondation Louis Vuitton, 2014).

As a project with a highly complex geometry, advanced design tools were required to handle it, such as BIM, cloud model, advanced parametric methods, and CNC processes.

Due to many different curved glasses (3500 molds), it was impossible to complete if not with automotive glass. Therefore, CNC was extensively used, and every extrusion was ordered from the BIM model. Furthermore, Digital Project (BIM) allowed the calculation of each panel and controlled the joint distances between them (Novakowski, 2012).

As reported by Novakowski, the core model consisted of a high-performance BIM master model, which served as the sole basis to support all the design phases. With the digital model, the team could develop studies in 3D, manage interfaces, identify errors in advance and directly manufacture industrial components with the geometric data.
The master model of the project had integrated all the BIM information within, such as finish specification, wall types and occupancy information, among others. Moreover, the high-performance BIM model was composed of different software, interoperable through standard formats and tools or web platform.
The project drew on the expertise from around the world and required to use a real-time, centralized BIM model server, to maintain the coherence of all authors and better communicate data (Novakowski, 2012). Moreover, the 3D BIM model will also benefit all the maintenance trades, including the visitors and curators, containing predictive information about material and life cycle of the facility.

Finally, the use of BIM in this project showed that BIM can be more than just a design tool; it is also a catalyst for a more collaborative design, energy efficiency, program analysis and fabrication.

### 3.3.2 Beekman Tower

The Beekman Tower building, located at 8 Spruce Street, Manhattan, was also designed by the renowned architect Frank Gehry. The building’s main features are the seventy-six-story high stainless steel wavy facades, placed on top of a six-story building with a brick facade.

The project was designed starting from the apartment’s layout, the elevator core and the zoning envelope regulations that the New York City Department of City Planning had imposed on the site.

![Beekman Tower project on 8 Spruce Street](www.urbanfile.org)

**Figure 25.** Beekman Tower project on 8 Spruce Street (source: www.urbanfile.org)

The facade with 39,738m², was covered with 10,911 pre-fabricated stainless steel panels. The group responsible for enveloping the building was PNA (Permasteelisa North America), with whom Gehry collaborated in all stages
of the design process, taking into account the budget and technical constraints (Goldklang, 2013).

PNA was chosen to be the envelope engineering consultant because of their previous experience with Gehry on several projects. Also because PNA was familiar with CATIA, a BIM tool, a variation of which is used in Gehry’s office and is called Digital Project (Goldklang, 2013).

PNA received the BIM model with all the panels, as created by Gehry’s office. In Gehry’s office, the use of BIM enabled the interactive substitution of double curved panels with single curved panels, detecting the effects of this shift in the façade.

![Figure 26. The analysis model of the curtain wall (source: Goldklang, 2013)](source: Goldklang, 2013)

The use of BIM technology in this project had several advantages. Since it was followed by continuous changes, the use of BIM allowed the automatic update of drawings according to the 3D model, which does not consist of just geometry as CAD does, but contains parametric objects with data attributes that allow various simulations and climate analyses to be executed.

BIM was also used to make the design process more collaborative to all the involved companies in the project, discuss changes, to identify in advance conflicts between the various systems of the building and to revise them.

In addition, since 9,023 out of 10,911 panels had a different shape, with traditional CAD would have been almost impossible to produce thousands of 2D drawings to manufacture the panels. On the other hand, using BIM was easier to do it by importing the 3D model directly to Computer-Aided Manufacturing (CAM) tools that operates with Computer Numerical Control (CNC) (Goldklang, 2013).

![Figure 27. Beekman Tower facade panels (source: www.arch-re-review.blogspot.ch)](source: www.arch-re-review.blogspot.ch)
CONCLUSION

Clearly, BIM technology has several advantages and it is considered a crucial tool for the AEC industry. The possibility to create and manipulate a digital construction model with all the information included on it, increased the efficiency and the consistency of the design process.

The Foundation Louis Vuitton project, clearly demonstrates that the use of the BIM approach was essential for the project success. Even though it was composed of design teams from all over the world, it allowed them to keep all the 3D model information updated. In addition, the BIM approach allowed the extraction of all the necessary information for the industrial fabrication. Furthermore, it also reduced errors in the construction by identifying them in advance.

In the second project, Beekman Tower, one of the major challenge was the design and the production of the irregular facade panels. They were adjusted with BIM tools and all the necessary information for their production was extracted from the BIM. Besides that, BIM allowed the project information to be always updated and used at any time by different building agents. Thus, making it a significant advantage for the project because of its constant modification feature.

The use of BIM technology in these projects also improved the collaboration between the construction agents, thereby increasing efficiency.

Finally, BIM tools create digital models of the buildings, from which we can generate plans, sections and views; we can simulate the structural details, external or internal interferences and performance analysis. Furthermore, each component generated from this digital model is linked to the project, generating a complete information by the end of the process.

Nevertheless, when we deal with more complex projects, that involves considerable data and irregular geometries, BIM tools are still not able to handle it. Therefore, an answer to these problems was given by Generative Design, as shown in the following chapter.
4 GENERATIVE DESIGN

4.1 INTRODUCTION

Despite years of development, CAD and BIM technology still fail when it comes to integrating the whole design process. The conceptual phase of design, in which the most important decisions of design (shape, material, structure) are taken, is usually not assisted by these software tools (Abrishami, et al., 2015).

The main reason is that the architect develop ideas through sketches and the current software does not offer any useful option. Moreover the 2D drawings and 3D models, are still laborious, what makes it a very raw and unintuitive process for the conceptual stage.

Furthermore, CAD and BIM tools are used just until a certain level of data complexity. In projects with more complex data, these tools are not able to solve problems, so there is a need to involve new tools capable of dealing with it.

As stated by Abrishami, et al, architecture is no longer just about the aesthetical emphasis anymore, but it is oriented on performance-based architecture, designed and tested with simulations, analysis, and optimization. Therefore, to deal with this constant growth of project complexity, the AEC industry was followed by many technological advances.

The transition of CAD to BIM technology, was a revolution in the AEC industry. BIM increased the speed, as well as the efficiency of project development, as well as generated the whole project documentation. Nonetheless, Hesselgren explains that there is a differentiation between the use of computers in architecture. All the CAD and BIM tools that were used untill now did not do else but just record design decisions. Moreover, recording design decisions surely is not the only thing that computation can offer.

An answer to these limitations was given by Generative Design (GD), a new approach to the design process that provides a more efficient use of computation in the creative process and design development.

4.2 GENERATIVE DESIGN DEFINITION

The generative design approach can be defined as a process based on rules or algorithms, through which various elements and design possibilities can
be created. These rules and algorithms are composed of parameters, which can be used systematically to generate various project solutions (Fasoulaki, 2008).

Conforming to Hesselgren, Generative Design is not about designing a building, but it is about designing the system that designs a building. In other words, we define rules and parameters and run it to generate solutions for us. Moreover, for this to work, the project must be decomposed into logical parts and relations first, to enable their transformation into parameters and algorithms.

GD approach is based in programming, so the parameters and rules must be encoded to enable the production of results. The more parameters and rules that are entered into the GD tool, the more sophisticated the results will be.

The number of generated solutions depends on the amount of data that the user introduces into the system. The less data entered, the bigger will be the spectrum of solutions because smaller are the rules that limit it. The following diagram shows the relation between data input and design development (Estkowski, 2013).

![Figure 28. Three ways of applying the generative design system (source: Estkowski, 2013).](image)

GD tools are computer applications that guide the form derivation with the evaluation of different alternatives. Through simulation techniques and optimization algorithms it seeks to achieve the intended performance (Batista Silveira dos Santos, 2009).

During the history of architecture, form was a major topic of discussion and after the introduction of GD systems its exploration acquired a greater importance. Therefore, it has motivated the development of a broad range of generative systems, also inspired by other disciplines, such as biology and mathematics. Among the most known Generative systems are Cellular Automata, L-systems, Voronoi Diagrams, Shape Grammars, Fractals and Genetic Algorithm (Fasoulaki, 2008).
4.3 GENERATIVE SYSTEMS

4.3.1 Cellular Automata

Cellular Automata (CA) is a generative system inspired by the biological growth and was first introduced to a model of self-production built by John Von Neumann in the 1940s (Silveira dos Santos, 2009). CA is a system of simple and identical components that together are capable of complex behavior. Such complex systems are found in nature, but their components are identical and simple, thus they are extremely relied in the cooperative effect (Wolfram, 1984).

![Graphs representing the sets of configurations generated in the initial steps of evolution according to a typical class 3 cellular automation rule (source: Wolfram, 1984)](image)

The CA components are cells organized in a grid, with only two states, empty or full (0 or 1). These cells consist of a set of rules connected that are related with the state of their neighbors. So, the Cellular automata do not require input from the user, since it is a self-referential process where the output of each generation becomes the input for the next one. Therefore, when we let a CA evolve, we can create very complex forms such as the ones in the Fig 30.

![Three-dimensionally mapped CA system variations (source: Knight, 2000)](image)
4.3.2 L-systems

Throughout history, architects have been fascinated by nature’s geometry, and many of them used these forms as an inspiration for their design. Therefore, within the last decades, we have developed a deeper study of these geometries.

In the 1960s, Arstid Lindenmayer developed algorithms that could model simplified plants. This system is called L-system, and it consists of four elements: initial configuration, set of rules, restrictions set and set of variables (Hansmeyer, 2015).

This generation of forms in L-system works by writing and rewriting the code, replacing elements of the initial configuration by others, thus transforming the initial model. This process is repeated several times and applies the conception of complex shapes with a form and a simple set of rules (Fasoulaki, 2008).

Figure 32 shows a project developed by Michael Hansmeyer to examine the possibilities of the L-system use in the field of architecture.

4.3.3 Voronoi Diagrams

Voronoi Diagrams have been studied for centuries, as evidenced by the studies of René Descartes in 1644. However, the one who actually defined these diagrams was Georgy Fedoseevich Voronoi in 1907. A Voronoi Diagram is a partitioning method of space into polygons that contain one point of generation (Fasoulaki, 2008). These polygons are created by connecting the closest points and then bisecting these lines with a perpendicular line.
The use of Voronoi Diagrams is very broad, from biology to city maps, and it has many benefits such as the organization of space based on the neighborhood, dynamic and interactive relationships and structural properties in 2D and 3D. In addition, the Figure 34 shows a study of a facade based on Voronoi Diagram cells (Bahraminejad & Babaki, 2014).

4.3.4 Shape Grammars

Shape Grammars are design-oriented generative systems that were first introduced by Stiny and Gips in 1972. They consist of a set of rules applied recursively to an initial shape. These rules contain basic spatial transformations such as translation, rotation, scale and reflection. This is performed in two steps, by recognition of the form and possible replacement. Therefore, it can generate an indefinite number of design solutions based on few rules (Fasoulaki, 2008).

A significant advantage of a shape grammar is the ability to recognize shapes that were not built into the grammar. So, it can also be used as an analytical system, decomposing projects into simple elements and reuse these elements to create another composition (Fig.35). These analytical systems can also be used to describe and analyze historical styles and existing languages of architectural projects.
This generative system has proven to be very efficient in the implementation of architectural intentions. An example of Grammar Shape usage was given by the enterprise Esri CityEngine, which through the software, CityEngine, call the corresponding Shape Grammar rules to create large scale models. These rules were responsible for creating the actual building geometries of the city of Pompeii (Fig. 36) based on the remaining buildings, paintings, and archeological data (CityEngine, 2015).

Figure 35. Ice-ray Grammar (source: Knight, 2000)

4.3.5 Fractals

Benoit B. Mandelbrot introduced Fractals in 1975, as mathematical rules that translate forms of natural shapes such as coastlines, snowflakes or clouds. Fractals are geometric forms that, when divided into parts, reproduce the original form in a reduced form. Thus, the subdivision of the whole process is
based in self-similarity form. Therefore, to establish a generative process based on fractals, it is necessary to establish an early form and a set of rules that replace each copy of the original form with a group of smaller copies (Fasoulaki, 2008).

Fractals have been used in architecture since the 80s by different architects as a generative system to create forms. The return of architectural interest in geometry and mathematics, and the use of digital technologies during the creative and development phase favored the usage of fractals in architecture projects (Sedrez, et al., 2014).

An example of fractal usage in architecture is the sun shade façade (Fig. 38) developed by Sadrez, Meneghel, and Celani, in their conference presentation at 32nd eCAADe (2014). The design process produced different alternatives for the brise-soleil using parametric definitions, with slight changes in the data. One of these alternatives was chosen to produce 120 different panels, which then were cut with a CNC machine (Sedrez, et al., 2014).

![Figure 37. Mandelbrot Set. (Source: www.math.utah.edu)](image)

4.3.6 Genetic Algorithms

The Genetic Algorithm is an evolutionary system, first introduced in the 1970s by Holland. This system has been used as a method of solving optimization problems of biological evolution. Based on the Darwinian
principle of survival and reproduction, GA transforms a set of individual objects that represent an initial population into a new generation. The initial population, called the genotype, is controlled by rules and properties embodied in the genes. This process is composed of four operations: the creation of the population, the selection, crossover, and mutation (Fasoulaki, 2008).

Selection searches for the fittest operators, so it duplicates structures with higher fitnesses and delete structures with lower fitnesses. Crossover recombines elements of the proper genotype from different genomes, to create better structures with better components. Mutation creates new structures that are similar to current structures with a small pre-specified probability.

The GA begins with a population of randomly generated structures, where each structure contains a solution to the task. It proceeds to evolve the generation, improving the structure in its current population through selection, crossover and mutation. However, GA does not guarantee the best solution because it has not clear a termination criteria (Fasoulaki, 2008).

The usage of GA in architecture was recently introduced to address with the complexity of architectural projects. It operates as a form generating tool and as an optimization tool to solve well-defined construction problems, such as structural, mechanical and performance based problems (Fasoulaki, 2007). An example of its usage in architecture is the performance-based exploration of facade designs (Fig.40), presented by Gagne and Andersen, in the Fourth National Conference of IBPSA (Gagne & Andersen, 2010).

![Figure 39. Crossover, in one of the fundamental mechanism of genetic algorithms (source: www.geos.ed.ac.uk)](image)

![Figure 40. Examples of GA facade exploration (source: Gagne & Andersen, 2010)](image)
4.4 CONCLUSION

The GD approach has completely transformed the role of computation in the design process. Currently, computation is not limited to just record the design process, but can also be used as a creative and auxiliary tool in design decision making.

The GD approach is defined as a process based on systematic application of rules or algorithm, to generate various project solutions. Therefore, the project results strongly depends in the data inputted in the GD methods.

Moreover, the above mentioned methods shows that it is possible to create many solutions and design alternatives with a certain level of autonomy. In addition, the process of working in GD basically consist of data inputs and model generation.

Nevertheless, as illustrated these methods generate only geometric entities. Therefore, the construction and design information are still not part of these tools.

Furthermore, since the results are generated by rules and parameters, often the final forms are impossible to build with current technology. That, because the GD approach itself is disconnected from the current reality of building technology.

The use of GD methods within a BIM environment can be the answer to this problem by making use of BIM construction elements and also allowing the extraction of the project information.

The part II of the present Thesis is built to demonstrate that the combination of these two paradigms can bring efficient and innovative results, and to conclude the importance of taking this new approach.
PART-II: GENERATIVE BIM TOOLS
IN THE DESIGN PROCESS
5 BIM WITH GENERATIVE DESIGN

5.1 INTRODUCTION
As stated in the sections above, BIM and GD, are two approaches that revolutionized the AEC industry. BIM introduced the concept of project information that increased the quality and efficiency of the design. In addition, the GD approach allowed the creation of sophisticated geometry by using algorithms, something that was impossible to achieve with other tools.

However, BIM and GD approach still have some disadvantages. BIM tools are being used only as a recording tool of design decisions and they work well only up to a limited level of project complexity. On the other hand, GD paradigm still has problems to identify and produce forms that are constructible with current technology of construction. Moreover, they do not generate any building information. Thereby, the AEC industry has two powerful tools, yet they are incomplete as separate entities.

An answer to this problem was given by Abrishami, et al, which proposes the development of a Generative BIM workspace. This approach is based upon GD methods that operate inside a BIM workspace allowing the design to be developed, monitored and controlled in a more efficient manner.

Generative BIM workspace, by combining BIM technology with computational methods, will help designers to solve complex design problems with creativity, fluidity and flexibility. In the following sections will be presented some related works and case studies that explore this approach.

5.2 STUDIES AND RELATED WORKS
5.2.1 DesignScript
DesignScript is a design tool based on programming approach and it was created by Robert Aish. Due to an increase in popularity of GD tools, such as Generative Components (GC) for Bentley and Grasshopper for Rhino, Autodesk saw an opportunity to create a product in order to compete with them. To that effect, they hired Robert Aish, who had previously worked on GC, and would best identify the market needs. The first objective was to create a textual language, but Autodesk insisted on a graphical language to be able to compete with Grasshopper (AEC Magazine, 2012).
Aish explains that DesignScript, as an associative language, maintains a graph of dependencies between variables that can be values or geometry. When DesignScript is running and we change a value using this variable graphic, it propagates that change to other variables that are affected by that variation.

DesignScript as a pedagogical tool is designed in a way that supports a very gradual approach to learning programming. But still, it is required a considerable amount of time and effort in order to be able to produce good results (Aish, 2011).

As stated by Aish, DesignScript aims to be more flexible because it is:

– focused on the end-user; introducing concepts that facilitate the users that aren’t accustomed to designing with programming. This is achieved by permitting the user to use its logical framework in order to produce the design models.

– multi-paradigm; introduce different programming paradigms in a single language.

– host-independent; geometry models are generated in various CAD applications and they can also access their various geometries, simulation libraries and create a correspondence between them.

– extensible; permits the users to add new tools and classes.

Currently, DesignScript is part of Dynamo, and it is available both as graphical nodes and as a textual language. With this integration, it became possible to run a project in Revit, a well-known BIM, or even run it as a standalone mode.
Furthermore, it is very accessible to new users that want to explore programming skills. For the non-programming users, it offers an approach to graph node diagramming similar to the Grasshopper, which is simple and requires no understanding of programming concepts. Therefore, the fact that DesignScript uses a graphical node diagramming similar to Grasshopper brings again the same problem of complexity and impossibility to make changes in our project.

5.2.2 Dynamo

Dynamo was developed by Autodesk, and it is a visual programming plug-in for Revit and Vasari. It offers a solution for the non-programmer designers, allowing them to use the Revit tool without the need of writing a single line of code.

Dynamo just as Grasshopper works with nodes and wires, where each node is a logical piece and it contains lines, Revit elements or mathematical functions. With Dynamo it is possible to directly access the Revit API, through Dynamo’s Python node, giving the user more flexibility to work with different methods, as well as to combine them, to achieve the final goal (Wong, 2015).

![Dynamo demo’s using a Stadium dataset (source: www.buildz.blogspot.ch)](image)

**Figure 42.** Dynamo demo’s using a Stadium dataset (source: www.buildz.blogspot.ch)

In addition, due to the fact that Revit is more adequate for the extraction of information, it is limited when it comes to complex modeling. However, when used with Dynamo it is possible to expand its limits.

Two of the main features of Dynamo are the possibility to automate processes and to customize the computational design tool. Wong explains that frequently we find ourselves developing simple logical systems that link together space, geometry, and function. For example, when we have a
project (Fig. 43) where the size of the office determines not only the number of furniture inputed in the room, but also the type of door used (locked/unlocked).

![Figure 43](image)

Figure 43. The furniture defines if the door is locked (source: www.case-inc.com/blog/revit-dynamo-more-than-grasshopper)

If this task is done manually, it will be tedious and prone to human error, because the task relies on manually counting the desks and also manually changing the doors for each area. With Dynamo, this task can be automated. Dynamo would automatically count the furniture loaded, measure the areas and determine the parameters of the door (Wong, 2015).

Finally, Dynamo is considered as a very accessible tool to new users that do not know programming and want to explore GD methods. Thus, it is being continually updated to offer better tools for the design development.

5.3 CASE STUDIES OF GENERATIVE DESIGN + BIM

The following case studies, Beijing National Aquatic Center and Aviva Stadium, were chosen due to their complexity and their need for sophisticated design tools. In addition, they illustrate the practical use of GD and BIM tools, and their efficiency on the design development.

5.3.1 Beijing National Aquatic Center

The Beijing National Aquatic Center or as it is often referred the ‘Water Cube’, was built for the 2008 Olympic Games. The project came as a result of an international competition in 2003 and the winning firms were PTW architect and the OVE Arup Consulting Engineer in partnership with the China State Construction and Engineering Corporation (CSCEC).
The concept of the building was inspired by water in its bubbly state and the square as the primal shape of the house in Chinese tradition. Due to the fact that, in Beijing, water sources are scarce, a building that symbolizes water has an unyielding presence to the nearby inhabitants. Moreover, the building works in duality with the complexity of structural elements and the simplicity of the whole building form (Eastman, et al., 2008).

The Water Cube plan is a square with 177m on each side, 31m height from the street-level, and it has 17,000 seats (11,000 temporary and 6,000 permanent). It contains five pools and an organically-shaped restaurant. After the Olympic games, it served as Beijing’s premier diversion centers.

The bubble shape covers the building complex structure with 100,000 m² of Ethylene-Tetrafluoroethylene (ETFE), a transparent envelope that weighs only 1% of glass’s weight. The result was 3000 irregular shapes of ETFE that rely on a complex structure of 22,000 stainless steel and 12,000 spherical steel nodes (Eastman, et al., 2008).

The building design team was also concerned with environmental issues. ETFE was chosen as the cover material because it permits the penetration of natural light and heat inside the building. Besides that, the solar rays that hit the building are 90% absorbed by the ETFE bubble and are reused to heat the pools and interior areas. Another important benefit is the fact that the roof of the Water Cube catches 80% of the water, which is recycled for the building needs.

The project was developed in three different phases: the design competition, the design development and the preparation of tender documents. In the

Figure 44. Water Cube ETFE facade (source: www.stylepark.com)

Figure 45. ETFE facade of Beijing National Aquatic Center (source: www.stylepark.com)

Figure 46. Sustainability issues (source: www.mumagi.net)
beginning, during the stage of the competition, the model structure was built with scripts in Microstation VBA. Then, it was exported to another CAD platform for the purposes of visualization and rapid prototyping. For visualization, ARUP developed a virtual model, which demanded the exportation of files from Microstation to Rhino (via IGES file) and then from Rhino into 3Dsmax, to generate AVI files, but that did not work well because it generated enormous files. Furthermore, for prototyping, these files were exported from 3D Microstation to an STL file, and then to DXF for structural analysis (Eastman, et al., 2008).

![Data transfers at the competition stage](image)

**Figure 47.** Data transfers at the competition stage (source: BIM Handbook 1st)

In the phase of the design development, the Microstation model was exported to Strand 7.0 for structural analyses and optimizations. Then, using VBA scripting, this model was exported to AutoCAD, Microstation Triforma drawings, and also to a Microsoft Excel (XLS) database (Eastman, et al., 2008).
In the phase of tendering, the structural dimensions were obtained by selecting sections of the wireframe analysis model, and importing them to Microstation. Then, for each structural element attributes were defined, such as section dimensions, reference numbers, and property information. This was done using VBA scripting. Finally, these elements were shaped by Triforma structural elements, to achieve the correct model (Eastman, et al., 2008).

As stated by Eastman, et al. (2008), the extraction of information and structural data was facilitated by Bentley Structural, which is a BIM software. Bentley Structural created an enormous library of the steel elements with their sections and their correct dimensions.

The extraction of project data, such as dimension, materials, and other construction information, was effectively generated by Bentley Structural. In order to document all the structural information, 112 well-detailed sections were used. Therefore, the main benefits of Bentley Structural utility were that it allowed the automatic documentation (plans, sections and schedules), the structural simulations and construction error minimization.

The fabrication of structural elements was made with on-site welding, with 3000 workers and over 100 welders. That was a unique practice for ARUP because usually the construction elements are made with prefabrication to avoid time loss and high costs. However, the client in China did not accept that. As a result, it was needed to generate 15,000 drawings for this non-standard structure to be welded, explained Eastman, et al. (2008).
The biggest challenges of the Water Cube were structural design and fabrication. The overall dimensions of the steel structure needed to be optimized, but also needed to fulfill the design requirement for the seismic situation in Beijing.

This process of optimization is followed by continuous changes to achieve the maximum benefits. Thereby, this optimization was constituted of 22,000 beams, in which should meet the 13 Chinese steel codes strength equations at 5 points on each beam for 190 load combinations. If these variables were multiplied, there would be 271.7 million design constraints to be optimized (Eastman, et al., 2008).

For this reason, Arup used Visual Basic 6.0 to write software from the beginning, which would control the entire optimization process. If this process were to be done manually, with simple software, it would be extremely laborious and would demand months (if not years) to achieve the best structural optimization (Eastman, et al., 2008).

Then for documentation purposes, a conversion program was needed to allow the transformation from the analytical software to a CAD software, and this process was done by ARUP, again using scripting.

This method was extremely useful for several reasons, among them:

- **It generated the full documentation of the project** (plans, sections, schedules, 3d Model),
- **Increased the modeling speed** (the whole model was built in 25 minutes), and
- **Improved visualization by exporting the models to 3D AutoCAD** (for the client and contractors).

One of the most important observations of this project was the use of scripting. As reported by the Arup Senior 3D Modeler Stuart Bull, the ability to use VBA scripts to create geometry, which links the analysis and engineering model to the working 3D CAD model, was irreplaceable, because then with the usage of Bentley Structural it was possible to extract all the 2D documentation and 3D models, for contractors and fabricators.

The positive impacts of BIM (Bentley Structural) are related to structural optimization, information exchange software, and interoperability, explains Eastman, et al. (2008). With BIM was also possible to rebuild the model daily or weekly, based on the structural analyses and optimizations, it also allowed the tender documents to be kept updated, avoid human errors, and also resolve other issues as sustainability, fire protection, and building performance.

Finally, this complex and unique project would be impossible without the strong commitment of the team, constant accurate collaboration, and specially the use of sophisticated design tools, such as BIM and GD methods.

### 5.3.2 Aviva Stadium

The Aviva Stadium was built in 1876, and it was the first international Rugby stadium in the world. Then, in 2007, a new project was redesigned by Populous in collaboration with Scott Tallon Walker, to replace the old structure and to meet the new present criteria.

The new design is characterized by a semi-transparent “shingled” organic skin, which wraps the entire stadium and the roof, covering all the seats and also providing natural light to the surrounding neighborhood.
The project team consisted of three groups: designers, project managers, and consultants. The design of Aviva had to take account of many different requirements, such as accommodating 50,000 seats, environmental issues, making the smallest footprint possible, being a worthy new landmark for the city, and also the ability to host international games.

According to Eastman, et al. (2011), for the conceptual stage, the designers concentrated on four criteria to develop the project:

- ensure the 50,000 seating capacity
- maximize sun exposure on the grass field
- minimize shadowing of neighboring houses (North)
- provide extra space for a training field and other auxiliary facilities on the east side of the site.

The result was a skin that covers the facade and the roof, with a complex system of trusses developed by Buro Huppodl. The height on one side where the neighboring houses are located is smaller, for the need to reduce shadows.

Figure 50. Aviva Stadium (source: www.info-stades.fr)

Figure 51. Series of daylight studies to evaluate different building forms and footprint. (source: Eastman, et al., 2011)
The design of the stadium consisted of a complex system of trusses, with different conditions such as shape, depth, element load bearing and the continuous refinement. All these parameters affected the structural performance of the structure. Therefore, using these parameters was possible to create a parametric model that facilitated the optimization of trusses and resolved several conflicts during the design stage (Eastman, et al., 2011).

Due to the constant change of the envelope, the team developed a collaborative workflow based on parametric models using Bentley’s Generative Components software. This tool enabled the control of geometry by numbers, which avoided the need to manually adjust the whole model.

The collaboration between teams started by setting some important rules about the project. The architects wanted to be able to adjust the external face of the façade, but surely inside the defined limits. Every change, made by architects in the envelope geometry, was automatically updated in the structure and also when the structure changed, the change was also reflected in the cladding panels and structure join pieces (Eastman, et al., 2011).

A graphic control was created to adjust the geometry of the model, and that helped the team to solve quickly revolve conflicts such as seating layout and stadium sightlines. In addition, a static geometry of tiers, with the appropriate spatial relationship was brought from Microstation CAD to Generative Component parametric model, which then was easily controlled with graphical parameters.

Furthermore, the firm Buro Happold needed structure analyses to optimize the structure, so they developed an Intern Application in Generative Component using the C# programming language. That connected Generative Component parametric model with Robot Millennium (a structural analysis software). This connection enabled the program to specify automatically dimensions, shape, truss and chords in the whole structure of the building (Eastman, et al., 2011).

Another important issue that was provided by the parametric model, was the process of fabrication and detailing, for both the envelope and roof structure. The output was a complete detailed spreadsheet with detailed sections.
In the Aviva Stadium project, both teams of engineering and design together, relied on several criteria to evaluate the design and constructability. The important lessons of this project are the benefits of parametric modeling for collaboration with architects and the great advantage of software customization (Eastman, et al., 2011).

To calculate double curvature and the complex truss structure in the traditional way would be extremely tedious, time-consuming and prone to error. By contrast, parametric modeling, programming, and customization permitted the teams to reach their goal with efficiency.

5.4 CONCLUSION
The above mentioned scripting tools, DesignScript and Dynamo, are the first attempts at connecting the potential of Generative Design approach with BIM tools. They have the ability to produce complex shapes, and deal with complex data, inside a BIM environment.

Even though these scripting tools are constantly being developed, they are still not enough to assist the designer’s work in an easier and an efficient way. In both textual and visual scripting, designers still have problems with the learning process as well as with the use of these scripting languages.

The Water Cube and Aviva Stadium, show that GD methods with BIM tools give us the ability to create highly complex shapes with great efficiency. In addition, the use of BIM allowed the entire project information to be stored and consulted at any stage of the design process. BIM also assisted in the production of details generated in the GD model, for fabrication and analysis.

Even though it is still at the beginning of its development, GD with BIM has proved to be a very powerful method in the AEC industry and it is being used in different projects all over the world.

In the following chapter we discuss the influence and possible changes of the use of GD methods to propose a new workflow in the design process.
6 NEW WORKFLOW IN THE DESIGN PROCESS

6.1 INTRODUCTION

In the previous chapters, this thesis described the development of digital tools and their influences in the AEC industry. In this section, we present a new conceptual workflow with the introduction of GD approach in the design process, as well as its more suitable use, based on our investigation.

The GD method was used in different complex projects, such as the previous examples of Water Cube and Aviva Stadium, where it has proven to be a very successful method. However, most designers still resist to include GD methods as fundamental design tools. This resistance also had happened before with the shift from CAD to BIM tools, and usually the cause is the long process of learning and lack of professionals that use correctly these new tools (Amorim, Souza & Lyrio, 2009).

The introduction of the GD methods in the design process depends on the type of approach to the problems that involved a large amount of information to deal with. If that information is available to be transformed into programming parameters, as much information as we introduce to the programming, the more complex are the results (Estkowski, 2013). This information can be dimensions, constraints or rules.

However, not all the project information can be translated into a programming language. Only some of the design problems have defined problem ‘domains’ that can be solved by the application of algorithms. Most of the design problems are interconnected with aesthetic and socio-cultural factors that cannot yet be systemized and subjected under the control of an algorithm (Arida, 2004).

Therefore, the GD approach is not always the right choice for the design process. It should be first evaluated based in each project individually, if it is the most appropriate approach because, as we saw in the BIM case studies above, some of the tasks are just easier to do with BIM technology. Thereby, in the following section we present some guidelines to select the most appropriate design tools.
6.2 THE MOST SUITABLE USE OF GD METHODS

The use of the GD approach allows us to explore and produce repeated elements and alternatives in design. Moreover, when these elements and data are generated by a variation throughout the project, as in the Water Cube case study, GD tools become crucial for the project success.

![Water Cube facade structure variation](source: www.architizer.com)

Nevertheless, when we design a project that is composed by simple elements and does not involve considerable complexity of data, such as the project of Municipal Library of Viana do Castelo, which consists of simple geometry with open spaces and opaque facade, it might be easier to execute it with the current BIM software.

![Municipal Library of Viana do Castelo](source: www.cm-viana-castelo.pt)

The following table sinthetyses the relation between the dimension and complexity envolved in a project and the choice of the most suitable use of the GD approach and BIM.
As seen in Table 1, we divide projects into complex and non-complex. The complexity of a project may come from the project program, from the external, legal or technical constraints; in its solution to the arrangement of spaces and articulation between them, in the search of shapes and volumes and in the use of geometric entities in the project.

Therefore, we can not define precisely the complexity in the origin but from observing the result. Consequently, we have to resort to the Rescher (1998) definitions of complexity. He explains that the complexity depends on the:

- length of the necessary amount of description
- length of the elements needed to produce a project
- time and effort involved to solve the problem
- number of elements in a system
- number of types of elements in a system
- different modes of the interrelationship between elements
- elaborateness of hierarchical relationships
- variety of modes of operation or functioning

- **Non-complex projects** - consist of a small number of simple elements, with a limited number of simple rules and data. These projects involve the fundamental needs such as extraction of drawing, modeling of simple shapes, visualization and management of data.

- **Complex projects** - consist of many elements, regular or not, with higher level of rules and data relationships. These projects also involve optimization, performance analyses and simulation.

Moreover, we divide projects into small scale and big scale projects. Currently there is no recognized definition that defines the project's scale.
Each project is unique and consists of its unique aspects, so it is difficult to set an exact line between what is small or large scale.

Still, there are some factors that indicate in which group the design best fits. These factors are: type of program, project duration, client and project cost.

The type of program can determine the scale of a project, in terms of: urban or architectonic, public or private, equipment or residential, individual or collective; and determines the physical and economic dimension of the project. Moreover, it also determine the number of interior compartments and the relations between them.

Then, the factor of the project duration is related to the time and number of phases that professionals need to design and perform the project.

Moreover, we have the client factor that also determine whether the work is large scale or small scale. In a small-scale project, customers are usually people with limited experience in design and construction, already in large-scale projects we usually work with client representations, which are specialized teams in constructions management.

Finally, the cost factor of the project also defines the scale of projects. Usually, in a small-scale project we have a lower cost with the construction, whereas with a large-scale project the costs are enormous. (The American Institute of Architects, 2004). But in some cases, a small-scale project is the one with the higher cost, even involving a lot of means to be built. It's the case of private houses that become customer or architect statements, or both. In most of these cases we have unique and reference architectures.

Finally, considering the scale and complexity factor of the project, our Table 1 results in a combination of four possibilities:

- **Small scale and non-complex** projects can be easily done in the current BIM software. A good example is the small private K-house(Fig. 54), which is based on an initial sketch of the designer’s idea, with basic parameters such as shape, dimensions and materials. This one demands the direct work of the designers with the design tool to achieve their intentions. This type of project is easier and faster to model manually in BIM software, rather than with programming tools.
- Small scale and complex projects are more efficiently executed by GD + BIM. An example that illustrates that is the project of Instat Houses developed by Marcel Botha and Lawrence D. Sass, using GD methods (Shape Grammar) to optimize the best and lowest cost solution, based on prefabricated elements, and then to extract information from BIM environment.

![KHouse Modern - the project evolution in Revit Architecture (BIM)](source: www.lifeofanarchitect.com)

![Use of GD methods to generate Instant Houses](source: www.caadria.org)
- **Big scale and non-complex project** can be achieved with BIM assisted eventually by Macros. If we have the process of designing a building that has many elements and tasks that repeat, that could be facilitated by using Macros. Macros are simple programs, that are more easily written by non-programmers, so they can be also used by designers with no programming skills, to increase the productivity, to eliminate tedious tasks and to gain more time for the creative process.

![Figure 56. Beck Technology Project developed on Revit Architecture.](source: www.slideshare.net)

- **Big scale and complex project** are very difficult to be done without the help of GD approach. The two projects mentioned above, Water Cube and Aviva Stadium, are examples that best illustrate the need for GD tools in this level of complexity and scale.

6.3 **NEW WORKFLOW IN THE DESIGN PROCESS**

The continuous changes in the design software technology also affect the design process. In the paper-based drawings, the designers worked with 2D representation and physical models. Then, in CAD, the designers began working with 2D and 3D representations. Thus, they needed to learn how to use CAD tools.

With the introduction of BIM technology, designers tried to learn how to efficiently use the digital sources (software) and managed to build geometric shapes until a certain level of complexity and also deal with a particular level of data complexity, allowing designers to produce 2D drawings and 3D virtual building models, simulations, tests and above all the extraction of design information. Hence, BIM approach also demanded that the design team should have people with skills in BIM tools.

Finally, the GD method requires working with programming. Since software consumes the major time of the design process, if architects do not know how to manipulate these softwares, they must have alternative collaborators.
in the programming area to create efficient solutions and metadata. With this kind of approach to the design process, the architect can now be monitoring the experiments by simulations, spending less time with software drawings and more time with the creative phases.

In the traditional process, i.e., all the processes that do not require an algorithmic approach, the designer works directly with the production of the project. So, he interacts with the design tools, generating 2D drawings, 3D BIM models, or physical models, to test ideas or solutions, as well as to develop the entire project.

Then, with the introduction of the GD approach to the design process, this has changed. Designers no longer interact directly with drawing or modeling tools in the project, but via GD methods. This fact opens discussion to how architects will develop their own projects since, according to Santos, Lopes and Leitão, some architects had never access to programming and there are others that had experienced some programming.

In practice some architects specialize in programming, but the majority only study programming up to a certain level. When they deal with projects that increase constantly in their data complexity, the challenge these architects face is so large that it will not pay off the programming effort invested on it. In this case, it is more productive to collaborate with software engineers (Santos, Lopes, and Leitão, 2012).

Therefore, architects do not need to become programming experts. They can master programming techniques up to the level that is sufficient for their immediate needs. Complex projects require a great amount of additional programming techniques and the effort needed for the architect to learn these techniques might not pay off. Thus, a good collaboration between architects and software engineers might be very useful (Santos, Lopes, and Leitão, 2012).

One complete project, besides architecture, is composed of several other specialties, such as structures, electric and hydraulic, and architects collaborate with the professionals of these areas to solve design problems. In addition, it is not surprising that the use of programming in the design process requires collaboration of the software engineers. The programming, in this case, would be one that interacts mainly with architecture and structure but, in the end, with all specialties involved.
Moreover, for this collaboration to work, architects and programmers should have some knowledge of each other area. The architect should learn at least basics of programming to know what to ask and what are the limitations of programming. Likewise, the programmer should have a geometry-based notion, 3D modeling, and some vocabulary of architecture, to be able to understand the architects need. This mutual knowledge grows with the experience of both teams during the different project development (Santos, Lopes, and Leitão, 2012).

This suggests the opportunity to start including in the project team the architectural programmer, a person that has knowledge of both areas, architecture and programming, the ones that ensure that communication between the conceptual designers and the software engineers is not misunderstood or interrupt.

Our following diagram shows the change in the traditional design process, as well as the introduction of the programmer in the process.

Figure 57. Workflow of Traditional and Generative Design process.
Nowadays, GD tools are usually being used only to solve problems and are not entirely integrated into the design process. They assist the traditional design process partially, but if they would be incorporated from the initial stage, as shown in the diagram above, we would achieve much more sophisticated projects (Programming Architecture, 2013).

During the development of a BIM project, it can also happen that the complexity of data involved in the project will increase, and it is necessary that the process moves from traditional to GD approach, and vice-versa. Therefore, the decision to work with traditional or GD methods can be taken separately for each task. Tasks can be isolated and worked partially, but it is preferable to design the whole project with one approach.

One of the advantages of working with just one approach is that it makes the adaptation easier. For example, if we design a building with BIM and its facade with GD methods, and there is a later adjustment in the BIM model size, the GD program must be re-executed and, possibly, changed, to regenerate the facade. If, on the other hand, the entire model was generated through GD methods, then any intended change in the model would need to be implemented through a corresponding change in the GD program and/or its parameters, but it would propagate automatically to all the building parts, thus maintaining consistency.

6.4 GENERATIVE DESIGN LIMITS

Generative Design is nowadays one of the most extensive and influential approaches to the design process. It offers an immense variety of possibilities and alternatives that are impossible to do with another tool. Nevertheless, Generative Design is not the tool that solves everything.

One of the most known abilities of GD approach is the possibility to identify the best project solution within a simulation or optimization test. Therefore, this works well only in the theoretical part; what happens in practice is that we can write optimization algorithms only for some certain concepts of the project. Nonetheless, we cannot easily write an algorithm that characterizes the aesthetic, design trend or socio-cultural environment of the place. Consequently, the program cannot optimize it.

Moreover, another limit of GD approach is the actual difficulty of writing a program. If we need to design a very complex building, it may not
compensate to do it all with programming. One of the main reasons for GD methods not to compensate is the cost of writing a program for a complex project, the cost of continued alteration and the cost of removing the bugs that come up as the program increase in size.

Furthermore, the project duration is a significant issue of AEC industry, and the GD method is not always the most recommended. If we are designing a 100-storey building, and we want to optimize the building based on different issues such as thermal, structure and building daylighting, the GD method will test a plethora of solutions. If these solutions take hours to run, it means that within weeks we will have less than a hundred solutions that are still not the best. Therefore, the algorithm would take still a long time to find the best model.

To conclude, we cannot throw away the knowledge of the architect and leave everything to algorithm, but surely the potential of the GD methods as design tools are enormous. The GD approach is still to be explored, as we will overcome the difficulties in terms of hardware and software, particularly regarding the learning curve of programming languages, the use and the ability to handle large-scale problems.

6.5 CONCLUSION
This chapter described how the integration of computational methods in the design process can assist the designer to deal with higher complexity of data and achieve better project performance.

GD methods are not supposed to replace the role of the designer, because as mentioned above, there are still many limitations of the GD approach. However, GD tools can assist the designer to achieve more sophisticated projects.

In the new reality, designers should work closely with professionals of programming to decide parameters, to monitor the project development and control the results. To this end, we conclude that GD methods only give efficient outcomes in an excellent collaboration between the designers and programming professionals.

This collaboration also depends on the each other knowledge of the architecture and programming area. The designer needs to know at least the
basics of programming to be able to talk to the programmer, and the programmer needs to have basic knowledge of architecture to understand the architects need. This fact is the reason for an emerging new profession: the programmer architect/designer as a member of a multidisciplinary team.

Finally, this thesis proposes a new design process using GD methods in BIM environments. In addition, it also presents some guidelines that help designer and programming professionals evaluate the complexity and the size of each project individually to identify the most suitable computational design tool.

In order to illustrate the above theories of the GD approach efficiency, in the next chapter we present a brief evaluation of it.
7 EVALUATION

7.1 INTRODUCTION

This dissertation aims to explore the potential of the GD approach as an auxiliary tool in the design process. In particular, it explores the influence of the GD approach and its capabilities to assist the project development.

Our proposal introduce a selection mode of the projects that best fits with GD methods and also a new workflow with the architects and programmers collaboration. We claim that the integration of GD methods on the design process, will improve the projects efficiency and their design development. In addition, that is already evident with various successfully built project around the world.

In this chapter we make a brief evaluation of the theory developed in this thesis. Our evaluation is based on two projects, Louvre Abu Dhabi and Absolute Tower. In the first project, we analyze the design process of Louvre Abu Dhabi, as a recent built project that heavily used GD methods. Then, in the second project, Absolute Tower, we make some simulations of the project modeling based on the two approaches, BIM and Generative Design. As a BIM tool we use Revit Architecture, and for Generative Design methods we use Rosetta, a programming environment for GD.

7.2 LOUVRE ABU DHABI

One of the most successful projects based on GD methods is the recently built museum, Louvre Abu Dhabi, designed by Jean Nouvel. It is built on Saadiyat Island, Abu Dhabi, with nearly 64,000m² and two-thirds of it is covered by a dome with 180 meters of diameter.

Figure 58. The Louvre Abu Dhabi: a major technical and architectural project by Jean Nouvel, based on BIM and GD methods (source: www.openbuildings.com)
The main feature of the project is the giant dome that performs environmental, aesthetic, and structural functions. The environmental function is to provide shadow and cooler temperature in the outdoor public areas that connects different museum buildings (Imbert, et al., 2012).

Aesthetic function of the dome is filtering the sun light, creating a dramatic and constantly transforming lighting effect beneath the dome.

The perforation of the dome was executed with computing methods by CERMA, an architectural and urban environment laboratory, in collaboration with architects to achieve their design intentions.

Moreover, the project was developed in a performative design approach, the inverse lighting paradigm in architecture, which works as mapping transparency tool. This map, is based explicitly in areas that designers hope to achieve light and temperature levels (Imbert, et al., 2012).
Moreover, the structure of the dome with its pattern, spans 165 meters between only 4 supports, generating large consoles. In order to meet these complex engineering and architectural requirements, the Buro Happold's SMART Sizer technologies, developed a structural optimization algorithm to define the interrelated constraints such as aesthetics, structure self-weight, cost and buildability. Thus, exploiting this approach enabled the production of efficient solutions which equally meets all the multi-objective design criteria. (Imbert, et al., 2012)

In addition, all these functions were translated into specific geometric rules for the design. Thus, the design team used advanced parametric tools and GD methods to find optimal solutions to shared challenges (Imbert, et al., 2012).

Therefore, at each level, the performance requirements were translated into geometric rules embedded in the core parametric model. With this, they had an integrated workflow, from the main geometry set down to the manufacturing data, in a single parametric model.

Thereon, with these simultaneous design processes, it became apparent the need for a shared parametric model. So, it was used Digital Project software and SVN, a web-based model repository, allowing the simultaneous work of architects, engineers and consultants in different project problems.
Moreover, the core parametric model was mainly developed without 2D drawings or 3D modeling, just with the rules embedded in the GD methods. They extracted 2D drawings or 3D models, just when it was requested for any other purpose.

Based on this example we can see that the GD methods are being very efficient when integrated in the design process. Thereby, the architectural intentions were clearly translated by the programming professionals, achieving complex and successful results.

7.3 **ABSOLUTE TOWER**

In order to test our previously mentioned theories, we made an experiment that better illustrates the potential of Generative Design methods. So, using the two most powerful design approaches, BIM and GD, we simulated the project modeling and some realistic changes, to measure their impact in time and effort.
The simulations were made by IST Architectural Master students: BIM approach was simulated by the author of this thesis, and GD approach by Sofia Feist, another student in Final process of the Thesis.

Our case study focused on the large residential buildings in Ontario, the Absolute Towers. These buildings are characterized by an unconventional shape, with their torsional form and dynamically fluid shape.

![Absolute Towers](source: www.urbantoronto.ca)

Figure 64. Absolute Tower exterior image and Plan (source: www.urbantoronto.ca)

Moreover, instead of developing all the details about this project, we concentrated on the core elements (structure, wall, floors and railing) of one of the Absolute Tower buildings.

The process of simulations was divided into two phases:

- The initial modeling
- The ongoing design changes

The first phase consists of the project modeling from scratch to measure the time and effort invested on it. In this stage, the programming approach needs more time to create parameters for each individual element of the project, as well as for their relationship. On the other hand, with BIM tools we already have most of the parameters included on the elements, so, we just start by using and connecting them.
As we can see in the Table 2, for the initial modeling, BIM tools are usually more efficient than the programming-based approach. However, the duration of the initial project modeling is not the most relevant if compared to the duration of future changes that are made on it.

In the second phase of this evaluation, we simulate some changes to measure the time and effort spent on it.

- The first change consisted of modifying all the slabs to a rectangle shape, with the external walls extended to the edges of the new slab form.
Figure 66. Absolute Tower model with rectangle shape floors (Revit Architecture/ Programming approach)

Table 3. Results of the time needed to change the floor into a square shape

<table>
<thead>
<tr>
<th>Change all the slabs to a rectangle shape.</th>
<th>Time(hours)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>BIM approach</td>
</tr>
<tr>
<td></td>
<td>1h 25 min</td>
</tr>
</tbody>
</table>

The process of change in the programming approach consisted of the time to modify the algorithm (2min), and the time to generate the model (8min), with a total of 10min.

On the other hand, with BIM approach it was needed to edit and adjust the new form of the floors and the railings. Moreover it was necessary to extend manually each wall that touches the edges of the slab, in each level of the building. So, all these changes were estimated to 1h 25 min of efficient work.

- The second change consisted of modifying all the slabs to an octagonal shape, with the external walls extended to the edges of the new slab form.
Figure 67. Absolute Tower model with octagonal shape floors (Revit Architecture/ Programming approach)

<table>
<thead>
<tr>
<th>Time(hours)</th>
<th>BIM approach</th>
<th>Programming-based approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>Change all the slabs to an octagonal shape.</td>
<td>1h 35 min</td>
<td>11 min</td>
</tr>
</tbody>
</table>

Table 4. Results of the time needed to change the floor into an octagonal shape.

The process of change in the programming approach is always the same but with different timing. The changes in this model consisted of the time to modify the algorithm (1 min), and the time to generate the model (10 min), with a total of 11 min.

On the other hand, with BIM approach it was needed to manually edit and adjust the new form of each floor and each railing. Moreover it was necessary to manually extend each wall that touches the edges of the slab, in each level of the building. So, all these changes were done into 1h 35 min.

- Then, in the 3rd example, we increase the number of floors, keeping everything else as it was, but following the rotation rule of 3°.
Figure 68. Absolute Tower model with more floors (80 levels) (Revit Architecture/ Programming approach)

<table>
<thead>
<tr>
<th></th>
<th>Time(hours)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Increase the number of floors to 80 levels.</td>
<td>48 min</td>
</tr>
</tbody>
</table>

Table 5. Results of the time needed to change the number of the floors.

In this case the programming approach changes were: the time to modify the algorithm (1 min), and the time to generate the model (6 min), with a total of 7 min.

On the other hand, with the BIM approach it was needed to copy the below floor, paste it in each new level and rotate it. In addition, all the external walls had to be updated to the new edges, changed as a result of slab rotation. All this work resulted in 48 min.

- Finally, the 4th change is to make the floors get bigger and bigger as the tower becomes higher, keeping everything else as it was, but extending the wall based on the new floor size.
Figure 69. Absolute Tower model with increasing size of floors (Revit Architecture/ Programming approach)

<table>
<thead>
<tr>
<th>Time(hours)</th>
<th>BIM approach</th>
<th>Programming-based approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>Change all the slabs to a square shape.</td>
<td>1h 52min</td>
<td>4 min</td>
</tr>
</tbody>
</table>

Table 6. Results of the time needed to change the size of floors in proportion with the level.

In this example the process of change with programming approach was very efficient if compared with BIM approach. The time to modify the algorithm (1min), and the time to generate the model (3min), with a total of 4 min.

On the other hand, with the BIM approach it was needed to copy the below floor and paste it in each new level. Then the floor was edited, scaled and rotated. In addition, all the external walls had to be updated to the new edges, changed as a result of slab rotation. So, this adjustment had to be done in each one of the 56 floors, and it was a hardworking and monotone task.

These simulations illustrate that with the use of GD approach, we can achieve more efficiency. Even though, in the initial modeling, BIM can be faster than GD approach, BIM still requires a great effort to make changes in the project.
The GD paradigm requires a huge initial investment in programming, but over the course of the design process, he gains in efficiency in relation to BIM.

7.4 CONCLUSION

The two projects above evaluated show the potential of GD tools in the design process. They illustrate the project changes and the exploration of project solutions, based on the GD approach. Moreover, they show how the use of the GD approach allow designers to explore more design solutions with more efficiency.

The first project, Louvre Abu Dhabi is a built example that shows how far GD approach can go. This project used algorithms to achieve the designers' intention. In addition, the structure and the environmental issues were also optimized using algorithms. Therefore, the success of the entire project was based on the GD methods and we conclude that the GD approach can go very far indeed.

The second evaluated example, The Absolute Towers, is a built project that we used for simulations. These simulations were done by modeling the building using BIM tools and GD methods, to compare the time spent working with each approach. Moreover, the modeling was divided in two parts, the initial modeling and post-modeling changes.

The initial modeling consisted of modeling the building from scratch on both BIM and GD approach. The BIM approach allowed to create the model using the own parametric elements such as wall, floor, pillar and railing. On the other hand, with the programming approach was needed to write the algorithm for all the elements and their interrelationship. As a result, in this case the modeling with BIM was much more efficient than the modeling with the programming approach.

In the second part, the post-modeling changes, we simulated some realistic changes in the same model of one of the Absolute Towers buildings, using BIM and GD approach. The 1st and 2nd scenario required the change of the floor format. Therefore, in BIM it was needed to adjust manually most of the building elements, which was very hardworking and tedious. On the other hand, using GD methods it was too easy to change the algorithm and it only took some minutes to generate the building. Thus, we consider that GD approach was very efficient to make those changes.
In the 3d scenario was needed to increase the number of levels, following the same rotate angle. Once again, with BIM it was needed to design manually all the new levels, while in GD approach it was needed only few changes in algorithm to execute again the building with those changes.

Finally, scenario 4 consist of constant increasing of the floor size, in each upper new level. This kind of modification showed that it is very difficult to be changed in BIM, but using GD methods it needed only few minutes.

From the observation of the evaluation results, we can confirm that the initial modeling may be faster in BIM approach, rather than in GD methods. However, in the course of the changes that occur during the design process, the GD approach is the most appropriate.
8 CONCLUSION AND FUTURE WORKS

8.1 CONCLUSION

Architecture is a process of continuous change and innovation, influenced by technology. Compared with the paper-based method, CAD was a revolution for the AEC industry, facilitating the project editing and increasing efficiency.

CAD applications allowed architects to deliver precise drawings that could be more easily altered without the need of manually deleting and redrawing all their plans. Nevertheless, these applications are far from addressing all the designer’s needs.

An answer to this problem was provided by scripting languages, which allow the manipulation of CAD tools. They access the underlying structure of existing CAD software and embed new functionality to it. Thereby, they can assist the design by simplifying repetitive tasks, by creating new features to the software, by performing tasks more efficiently and by manipulating large amounts of data.

Still, errors are liable to happen, and CAD programs do not have the capacity to recognize them due to the inability of interconnecting different inputs in a project. Thus, changing a complex project is still a hardworking and prolonged task. Another disadvantage is that these drawing and models are pure geometry with no construction data, causing documentation problem.

The introduction of BIM technology exceeded the efficiency of CAD, allowing the modeling of virtual construction reality and the production of all design information, made possible by the use of parametric BIM objects containing all the necessary information.

This information can be extracted from the BIM model at any time during the design process, consisting of plans, materials, dimensions, cost estimates, project schedules and analyses. Moreover, it serves also to support all the involved parties in the design and construction processes, from contractors, to builders, to clients, and even the post-construction maintenance staff.

Even though BIM is defined as a powerful approach to the designing process, it is unable to solve complex projects that involves complex data and irregular geometries.
Therefore, an answer to these problems was addressed by Generative Design approach, which allowed designers to explore a side of geometry and data complexity that was not possible before.

The Generative Design approach is defined as a process based on rules or algorithms, through which various elements and design possibilities can be created. These rules and algorithms are composed of parameters, which can be used systematically to generate various project solutions.

Nonetheless, as illustrated in this thesis these methods generate only geometric entities, however lacking construction and design information. In addition, due to results being generated by rules and parameters, often the final forms are impossible to build with the current technology. This being so due to the GD approach disconnectedness from the current reality of building technology.

The use of GD methods within a BIM environment is the answer to this problem by making use of BIM construction elements and also by allowing the extraction of the project information.

The Water Cube and Aviva Stadium project, showed that GD methods provide the ability to create highly complex shapes with great efficiency. Furthermore, the use of BIM allowed the entire project information to be stored, consulted and extracted at any stage of the design process.

Even though at its initial stage of development, GD with BIM have proved to be a very powerful method in the AEC industry and are being used in different projects all over the world. However, the GD methods are usually not integrated in the design process. They are used partially, only when needed to solve complex design problems.

This dissertation argues for one main point: integrating Generative Design approach in the design process to simplify the handling of changes and data complexity. In particular, we propose the use of GD methods within a BIM environment.

Our proposal presents a new workflow that integrate GD methods in the design process. Moreover, we also describes the role of the designer in this new workflow as well as the influences in the process of design.

GD methods require programming knowledge which most of the designers lack. Therefore, we concluded that in the new reality designers should work
closely with professionals of programming to decide parameters, to monitor
the project development and control the results. To this end, we also outline
that designers should have some basic knowledge of programming to
execute simple tasks for their own needs. If there is a very complex task,
then they should invite programming professionals to deal with it.

In addition, this thesis also come up with some guidelines that assist
designers and programming professionals to identify the most suitable
computational design tool for the project. Thus, by evaluating the complexity
and the size of each project individually.

Finally, to illustrate the above theories of the GD approach, an analysis of a
recent complex built project, Louvre Abu Dhabi presents the highlighted
issues and benefits of integrating the GD approach. It proved that using
algorithms was possible to transform the structural, environmental and
design intentions into a project solution.

In addition, we also evaluated the formalization of another built project,
Absolute Towers, using the two most powerful design approaches, BIM and
GD. In this evaluation, we simulated the initial project modeling and some
realistic changes, to measure their impact in time and effort.

From the observation of the evaluation, we can confirm our main topic of
this thesis that the GD approach can assist designers to efficiently handle
changes and explore solutions. If the designers master the GD approach, they
gain conceptual freedom to make changes, enhancing the project and the
searching for a high performance design.

8.2  FUTURE WORK

Future work should improve the evaluation and the development of the
algorithmic approach, integrated in the design process. Some topics that
might complement this thesis are:

1. We developed a table that assist designers and programming professionals
to choose the most suitable design tools, based on the size and project
complexity. However, it would be interesting to produce a simulation for
each proposal to better illustrate it.

2. Additionally, we proposed a new workflow within the GD approach in the
design process. Still, it is necessary to detail it, including all the other building
agents in the process, to better understand how the GD approach affects the AEC industry.

3. The evaluation that was produced in this thesis consists of a simplified modeling to test initial changes in the design process. However, it would be interesting to improve the formalization of the project up to advanced phases. More construction elements should be introduced in the model to evaluate the limits of this approach.

4. Another evaluation that would enhance this dissertation is the cost calculation of each approach, BIM and GD in different projects. Thus, it would be even clearer, when GD approach do really compensate.

Shortly, future works should try to reinforce arguments that the GD approach is the right tool for improving efficiency and handling project changes.
BIBLIOGRAPHY


36. Programming Architecture - What is your profession? (2013) (video file), Available at: https://www.youtube.com/watch?v=jwRPQrxxNDA.


