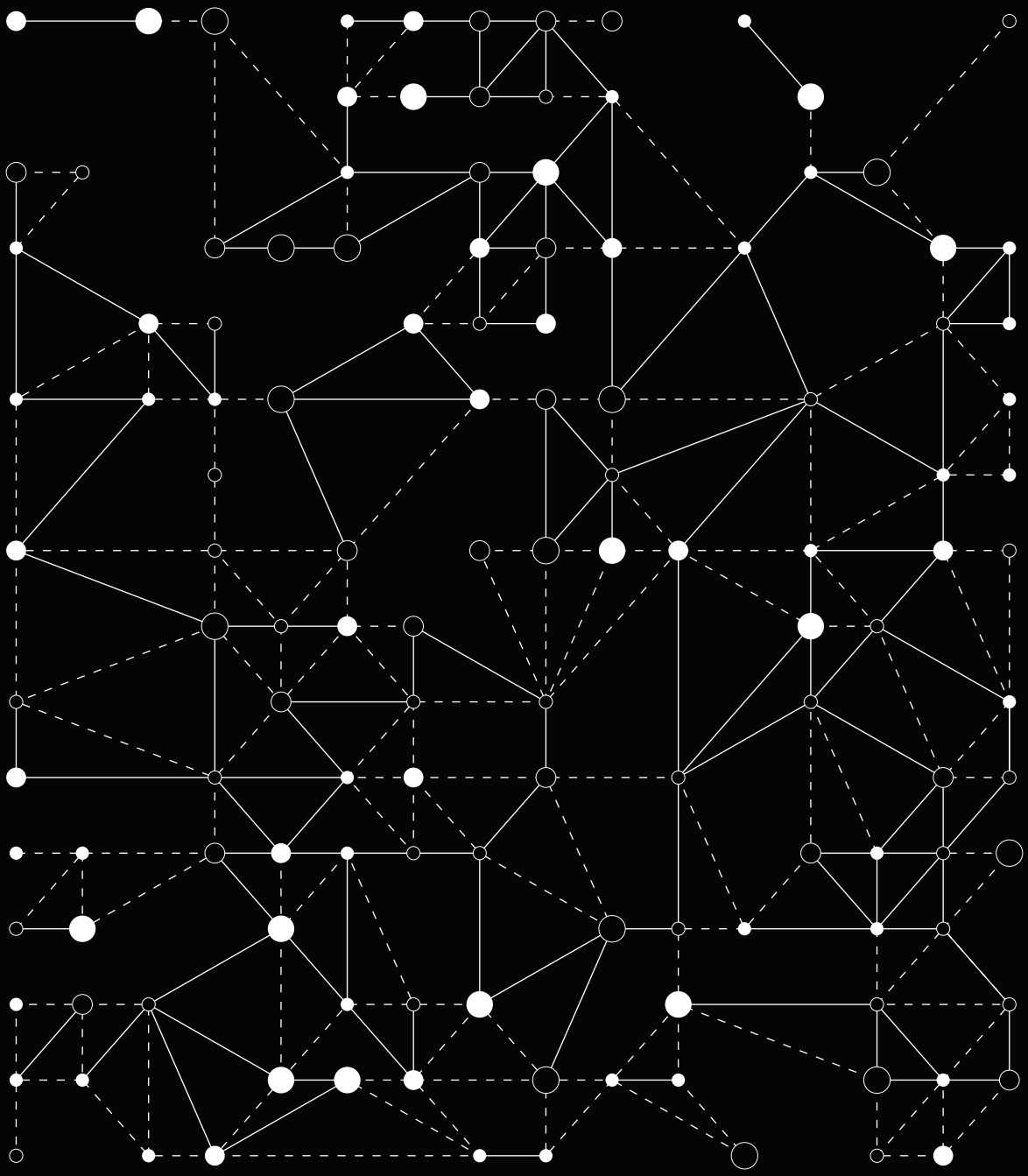
GENERATIVE AGENT-BASED DESIGN

TH-OWL UNIVERSITY OF APPLIED SCIENCE AND ARTS - MID/CD – 2019
PAVEL FURTSEV



June 2019

TH-OWL University of Applied Science and Arts

Master Integrated Design - Computational Design

Pavel Furtsev (author)

p.furtsev@gmail.com

Matriculation №: 15399078

Prof. Dipl.-Ing.

Jens-Uwe Schulz (first supervisor)

jens-uwe.schulz@th-owl.de

Prof. Dipl.-Ing.

Hans Sachs (second supervisor)

hans.sachs@th-owl.de

To my family

ABSTRACT

Generative agent-based design is promising to provide a solution for complex design problems in architecture. This approach is addressing architecture as a complex system and creates emergent solutions. However, the field of generative agent-based design is relatively new and lacks the answers to the most fundamental questions. In this research, it is proposed that agent-based generative design is an instrumental part of the "new paradigm" in architecture. We give a general definition of the term and formulate the building blocks of the generative process with "General reference model for agent-based modeling and simulation" (GRAMS) as the main component.

TABLE OF CONTENTS

LIST OF FIGURES

INTRODUCTION

MOTIVATION

We are now at the time of big opportunities and great challenges for architects and designers. Opportunities are brought by new advanced technologies that blend into architectural design, engineering, and construction. The challenges we face in the developing world, are emerging from global sustainability problems, urbanization, and population growth. The issue of contemporary architecture is that it is disconnected from contextual surroundings, nature, and society [1]. We have to utilize the new technologies and developments to be able to tackle all the problems in an integrated manner.

Implementation of Generative agent-based design (GAD) as an architectural design method and tool can bring more possibilities to create better, optimal solutions. The field of computational design is relatively new, but it is already giving solutions that were not possible before. The agent-based method in generative design is capable to produce outcomes that are even more comprehensive, connected and emergent. However, the concept of using agent-based modeling for generative design is new. The answers to the most fundamental questions in the field are still missing. The definitions are not clear and biased or not generally accepted. The tools for GAD are constrained by a few approaches. It is necessary to research further on the topic to answer the most fundamental questions of agent-based generative design.

OBJECTIVES

> *"The most creative ideas ever experienced are often conceptualized by asking simple questions"*
>
> *JEFF SHINABARGER*

Main goal of the thesis is to push forward generative-agent based design as a methodology and technique to create better solutions in computational architectural design. We take generative agent-based design not as a new philosophy of design, but rather as practical approach to search for design solutions. We aim to bring up the basic concepts of generative agent-based design as a process, its current problems, and to suggest solutions to fully accumulate the potentials of agent-based design. The objectives of this thesis can be formulated in three simple questions:

1. WHAT?

What is generative agent-based design: terminology difficulties and clarifications? What is the state of the art of generative agent-based design in architecture?

2. WHY?

Why should we use the generative agent-based design? Why today it stands out from other methods? What is the potential of GAD?

2. HOW?

How to perform generative agent-based design? What are the issues and how to solve them?

1. WHAT ?

What is generative agent-based design? The answer to the question is given through investigations of its ***terminology*** and meaning, as well as from the research of the ***state of the art*** in the field. It will also illustrate the focus and boundaries of this master thesis research.

1.1. TERMINOLOGY

Even though the term exists in the field for quite a while, surprisingly there is no generally accepted definition of *generative agent-based design*. To clearly understand what we are dealing with, we need to give an extensive answer to what generative agent-based design is. Precise terminology is important for coherent comprehension of further discussed questions. The strategy we adopt here is deductive reasoning. The term can be dissected in two main concepts: generative design and agent-based approach. Further, we will put these terms back together for a conclusive definition.

GENERATIVE DESIGN

Common explanations of this term are usually not general and in most cases describe a specific approach or goal of generative design. For example, "shape grammars", "genetic algorithms" or "topology optimization". Anthony Hauck in the article "What is Generative Design?" gives the definition of the term that seems to be the most convenient [3]:

> *"**Generative design** is the automated algorithmic combination of goals and constraints to reveal solutions"*

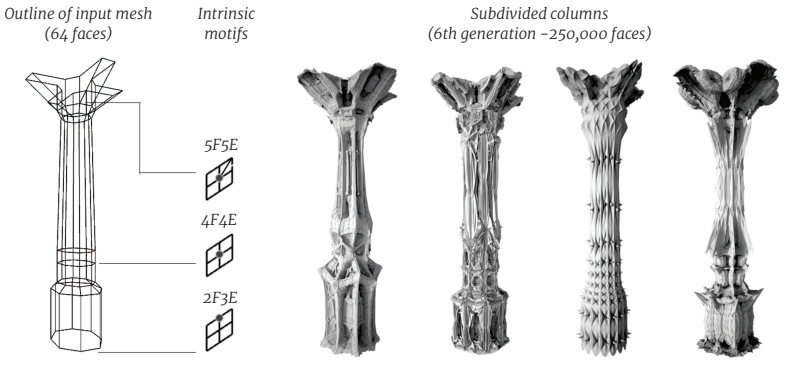


Figure 1:
A column generated through algorithmic subdivision [2]

If we think about generative design as a rule sequence or algorithm based design, this concept is not new. Design with manual algorithms or rules dates back to work of Vitruvius in I century BC [4], as well as Andrea Palladio in the XVI century [5]. More recent (1977) related work is *"A Pattern Language"* by Christopher Alexander [6], which addresses design problems with a pattern (pattern not in a decorative sense but in a sense of a repeated, common design solution). These patterns create a design language and are capable to guide in manually generating an appropriate design.

"Automated algorithmic" is stated in the definition to emphasize the fact that this process is implemented with the help of computational technologies, to distinguish between the similar ideas by Christopher Alexander, Palladio, and Vitruvius. Algorithms are created to reach a certain *"goal"* in a range of specified *"constraints"*. The result of an algorithm are solutions to a given problem. The definition uses a plural version *"solutions"* to emphasize the fact that by automating the process we can produce multiple solutions to choose from.

AGENT-BASED

Because of the interdisciplinary use of terms based on the concept of agents, there is a lot of complication and confusion. However, there is major agreement on the definition of agent:

> *An **agent** is an entity that is situated in some environment, and that is capable of autonomous action in this environment in order to meet its objectives [7].*

The agent is the core of concepts from the computational fields like Agent-Based Modeling and Simulation, Multi-agent Systems. These concepts are related and very similar, however they are distinct.

> *An **agent-based model** (ABM) is a simulation model that employs the idea of multiple agents situated and acting in a common environment as central modeling paradigm [7].*

> *A **multi-agent system** (MAS) is a computerized system composed of multiple interacting intelligent agents [8].*

Figure 2:
Simulation of flocking birds with ABM [12]

Figure 3:
A photo of flocking birds [13]

Both terms, MAS and ABM, overlap and sometimes used interchangeably. However, there are distinctions that we have to keep in mind. The main difference is that in MAS the agents are intelligent, they can control their own local subsystem. Intelligence may include methodic, functional, procedural approaches, algorithmic search or reinforcement learning. Such agents do not have all data or all resources needed to achieve an objective and need to collaborate with other agents. In this case, data is decentralized and execution is asynchronous. MAS is more related to Distributed Artificial Intelligence (DAI). On the other hand, ABMs are focused to search for explanatory insight into the collective behavior of agents. In this case, the agents are not necessarily intelligent, they just have to fallow a simple set of rules to achieve an emergent behavior [8–11].

GENERATIVE AGENT-BASED DESIGN

To sum up the term, we cannot just take two definitions and merge the thoughtless. Even though mentioned definition of generative design seems to be clear and objective at first glance, it is puzzling and could be simplified. Saying *"combination of goals and constrains to reveal solutions"* is not necessary here. When we say algorithmic and automated it implies that it is a combination of goals and constrains and the fact that it is a generative *"design"* implies that it is focused on finding design solutions. Saying *"process"* would keep the definition more general and simple. It creates a focus on the fact that a designer is actually creating a process based on automated algorithms that will create the final design options:

> ***Generative design*** *is the automated algorithmic design process.*

In generative agent-based design, *"agent-based"* is a clarification of a type of the process that is used to create the designs. Generative agent-based design can include both ideas, MAS and ABM, as a generating process, which means that the focus is on the concept of agent in general but not on the method. This can be interpreted in general definition of generative agent-based design:

> ***Generative agent-based design*** *(GAD) is the automated algorithmic design process based on multiple interacting agents.*

STATE OF THE ART 1.2.

Agent-based modeling and simulation concept was present since the 1940s, despite that it did not acquire a widespread implementation until a burst of computational power and maturity of conceptual grounds of agent-based simulation in 1990s. Today the use of ABM is significantly widespread. The applications of agent-based modeling and simulation range from biology (analysis of the spread of epidemics, population dynamics) to economics and social science (financial market simulation and prediction, cognitive social simulation) [9].

The ABM for design purposes was first implemented in urban design. The fact that urban studies are strongly related to social aspects and require an intensive analysis of social behavior, created the need for relevant simulation tools and methods. ABM has a series of advantages over conventional modeling paradigms, which makes it particularly suitable in socio-related studies, thus in urban analysis and urban design in general. First examples of implementation include geospatial analysis, social segregation and aggregation analysis and residential dynamics analysis, simulation of size-frequency distribution of traffic jams [14]. Today with rapid urban growth and a concept of "Smart City" ABMs are extensively used on various levels of urban planning and urban transportation design [15-16]. The use of agent-based modeling here is mostly for analytical, optimization and design decision-making purposes.

Figure 4:
Study of the pedestrian movement patterns in urban environment [17]

In architecture, agent-based modeling is used as a means of both, *conceptual paradigm* and as an *instrumental device*. Since we are researching on *"generative"* agent-based design, it is important to note that the focus is on the generative qualities of the ABM. Thus, we are interested more in the methodological and instrumental implementation of the agent-based design in architecture. Current design and research projects explore GAD from different perspectives and introduce various objectives and applications by means of different methodologies and tools. In most cases, the design objectives are coupled or integrate the possibility of interactive design. Literature review shows that the most recent design and research projects are using agent-based modeling for:

- ***form-finding*** [18, 19]
- ***environmental performance*** [18, 20, 21]
- ***structural performance*** [20, 22, 23]
- ***computational morphogenesis*** [24-26]
- ***spatial organization*** [23,27]

To reach the design objectives diverse methods and approaches are used. Some outstanding selected projects are described further in more details in terms of the focus and objectives of the design and techniques used to creating agent-based or multi-agent systems.

RESEARCH OF INSTITUTE FOR COMPUTATIONAL DESIGN AND CONSTRUCTION, UNIVERSITÄT STUTTGART [24-26]

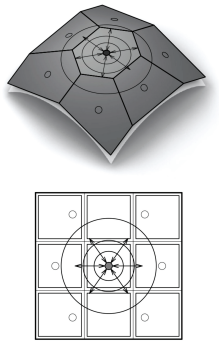


Figure 5:
Motion Behavior mechanisms (the attraction and repulsion steering behaviors)

This research investigates the development of an agent-based design method as an integrative design tool for polygonal surface structures. Integration of material, fabrication and construction constrains here referred to as computational morphogenesis, hence the generative agent-based design as an approach for it. The geometrical properties of the plates, which inform the behavior of agents, are inspired by the sea urchin and constrained by the fabrication limitations.

In their work, Baharlou and Menges introduce the Constrained Generating Procedures (CGPs) as the main technique for generative agent-based models and drivers of emergent behavior of the system. CGPs are generating and constraining mechanisms of the agent-based model introduced by Holland . In this particular project, motion behavior mechanisms and geometrical behavior mechanisms are present.

The motion behavior mechanisms are based on the autonomous motion rules developed by Reynolds –. The geometrical behavior mechanisms and constrains are directly connected to material properties, fabrication and construction limitations. Since the material is constraint to a flat plane geometry, the geometrical behaviors include tangent plane intersection mechanisms. The logic of agent system includes agent division and growth mechanisms. To simplify the complexity of behavioral system the aggregation techniques are applied.

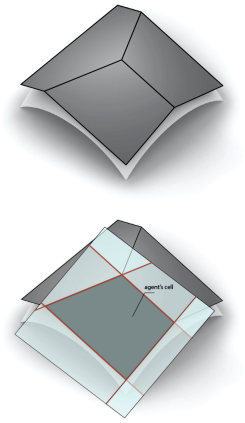


Figure 6:
bottom: The generating agents' cell;
top: The intersection mechanism with slicing algorithm

RESEARCH OF UNIVERSITY OF SOUTHERN CALIFORNIA
[11,19,20,30]

The focus of this research is broad but it concentrates in two case studies. One is the reciprocal shell structures and another one is façade envelope panels. The main objective within those two case studies is to develop a multi-agent design framework, which is integrating several design problems. Addressed design problems include: design generation and evaluation, environmental and structural performance, fabrication aware form finding.

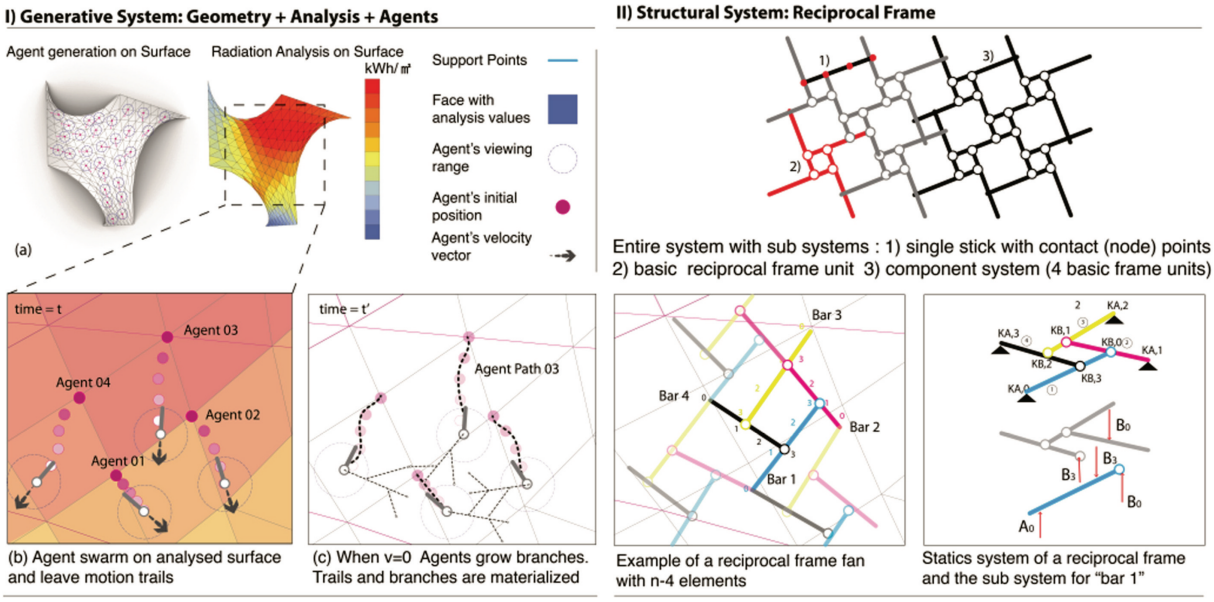


Figure 7:
Shell generative and structural systems with agent behaviors

The multi-agent framework consists of various custom agent classes and their interdependencies. Each design cycle is implemented within different agents. The framework consists of two stages. First, is intended for generative aspect of design, where agents act autonomously. Second, is intended for optimization of the design, where agents act collaboratively to find optimal solution. Agent classes include:

- Generative agent
- Specialist agent
- Simulation agent
- Evaluation agent
- Coordination agent

All agent types have the same number of layers, but their definition is based upon a specific class. Behavioral rules may vary in complexity and levels of information taken into account during the decision-making process.

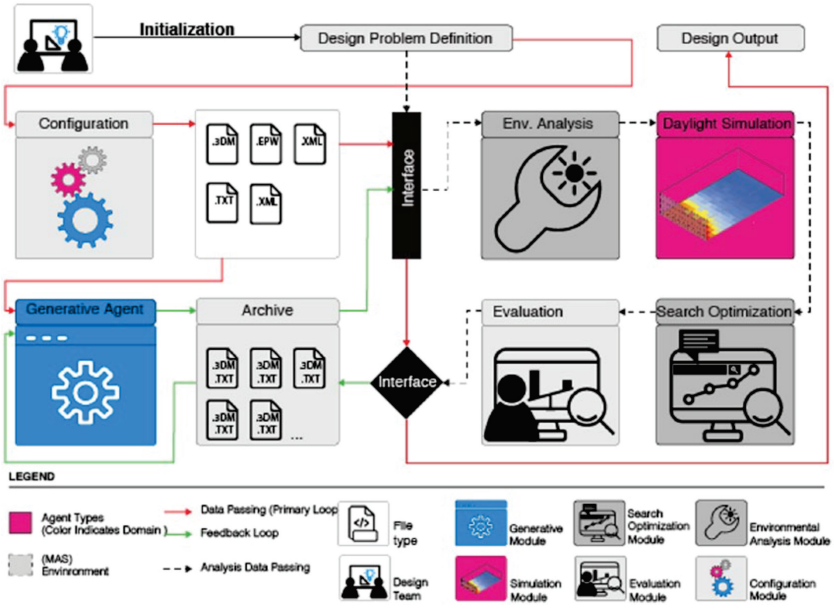


Figure 8:
Multi-agent system framework

SOFTWARE TOOLS

Majority of the studied projects use custom algorithms within existing software packages or even complete custom developed software. In some cases, the software is used in different combinations depending on the design process and design objectives of the project. Two most commonly used and available tools include *Processing* and *Rhino-Grasshopper*.

Figure 9:
Composite Wing – robotically fabricated inlay within a composite surface [33]

PROCESSING

Processing is a programming language and development environment used for teaching fundamentals of computer programming within a visual context. It serves as a sketchbook and as a production tool for artists and designers [31]. Designers are using Processing and various libraries to create custom algorithms for generative design purposes, which includes also the agent-based algorithms. *Figure 9* shows one of the projects by Roland Snooks generated in Processing with agent-based algorithms.

Another case for using Processing for agent-based design is the work by Satoru Sugihara. He developed an algorithm development environment named iGeo, which is an open source library for Processing [32]. Example of project developed using iGeo and Processing are shown in *Figure 10*.

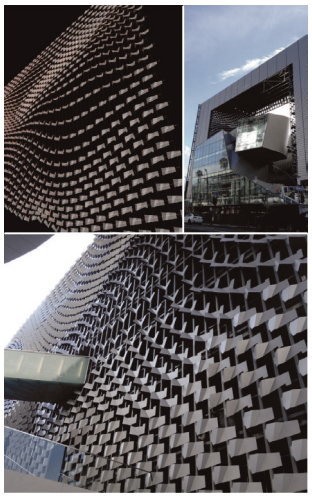


Figure 10:
Formation of folded panels at Emerson College Los Angeles Center [32]

RHINO-GRASSHOPPER

One of the most common software for generative design in architecture is combination of Rhino 3D NURBS modeling software and Grasshopper visual programming plugin. Grasshopper includes a huge variety of tools and a few ones for agent-based design. Custom algorithms can be created in different programming languages like Python, C#, Visual Basic. Some add-ons for Grasshopper include tools for simulating agent behavior based on swarms (QUELEA, NURSERY, PHYSAREALM). Projects which utilize Rhino-Grasshopper tools are [18-21,25].

2. WHY ?

Now we have an idea about what is generative agent-based design, but we have to justify and explain the reasons ***why*** we should use it. To give the answer we will look back to where the concept of GAD is coming from and what are the potentials of the method.

2.1. "NEW PARADIGM"

Today we are witnessing a great change of conceptual grounds in architecture. On the one hand, this change is said to be driven by emergent digital tools and the ability to work intensively on the topology [34]. On the other hand, the paradigm shift is driven by the Complexity theory as a new worldview. It opens the possibility to develop new design approaches and strategies for architecture where a designed object is seen as a complex system [35]. The "new paradigm" is the answer to the question "Where the idea is coming from?"

Paradigm is a distinct set of concepts or thought patterns, including theories, research methods, postulates, and standards for what constitutes legitimate contributions to a field [36]. The title of the chapter is quoted for the reason that it is still a big question, which falls out of this thesis work margins if we can call it a new paradigm. The field of architecture is still substantially modernistic and the "new paradigm" is more an architectural movement that has the potential to change architecture dramatically. Here a "new paradigm" is more the focus on an instrumental approach that is developing out of the concepts coming from Digital Revolution and Complex systems theory.

COMPLEX SYSTEMS THEORY

> *"The whole is more than the sum of its parts"*
>
> *ARISTOTLE*

From the middle of the 20th century, a major change happened in natural sciences, which was outlined in Complex systems theory. As a study of complexity and complex systems, it is focused on relationships of elements and their self-organization. Theory of complexity is opposed to the linear and mechanistic Newtonian worldview.

Figure 11 shows that the science of complexity has been developing as an interdisciplinary knowledge base of concepts and theories. With time complexity theory has evolved from describing properties of specific given systems into a broad area of research in science with applications ranging from economics to physics [35]. Currently, there is no consensual general definition of complexity and no unified theory [37]. Yet, it is possible to outline the main ideas that build up the concept of a complex system.

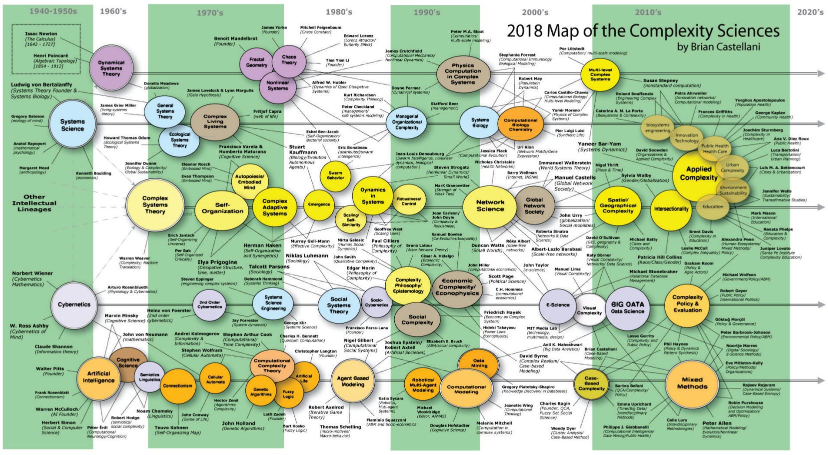


Figure 11:
Map of Complexity Science [38]

WHAT IS A COMPLEX SYSTEM?

The word "complex" here implies diversity, through a great number, and wide variety of interdependent, yet autonomous parts. The "system" portion refers to a set of connected, interdependent parts or a network [39]. A system is a collection of interacting elements making up a whole. It is important to distinguish that the system can be complicated but it does not mean that it is complex. To be considered a complex system it has to have some essential properties [40]. Many natural systems possess the properties of complex systems that could be explained on a very common example, an ant colony [41]:

- *Complex systems consist of a large number of* ***interacting elements***:
  An ant colony usually consists of a great number of ants with different functions like foraging, nest maintenance, patrolling, etc. Ants interact and communicate with each other through tactile and pheromones.

- *Complex systems are **decentralized***:
  The behavior of the colony depends on the interaction between each ant. Ant quine does not give orders or control the colony. Usually, her function is to reproduce.

- *Complex systems are **open**. They can have external (outside of the system) interactions:*
  The size and structure of the ant colony depends on the environment where it is situated. We can never isolate the existence of the colony from its environment. Through the interaction with the ecosystem where it is situated it is adopting and changing.

- The relations in complex systems are ***nonlinear*** and ***dynamic***:
  An ant colony is never static, the number of ants is always changing, the nest is constantly modified and rebuild. These changes are reliant on the variety of environmental conditions and on the current state of the colony.

- *Complex systems exhibit **emergent** and **self-organization** behavior:*
  Ants decide on which function they perform in the colony depending on the number of other ants they communicate with. This results in decentralized, self-organization of the colony and emerges in a cooperative intelligence.

MODELING AND SIMULATION OF COMPLEX SYSTEMS

The practical benefit of knowledge about the complex systems and other concepts of complex systems theory mainly lays in scientific modeling. The aim of scientific modeling is to make a particular part or feature of the world easier to understand, define, quantify, visualize, or simulate. It requires selecting and identifying relevant aspects of a situation in the real world and then using different types of models for different aims [42].

The modeling of complex systems present some difficulties related to its properties. Since complex systems are nonlinear and result in emergent behavior, it is not possible to predict or comprehend the system just from studying the parts of the system and then combining the results. This created a trend to model systems in detail and then simulate their behavior, which is called microscopic modeling of systems.

Macroscopic modeling of a complex system focuses on describing the aggregate behavior of a system and usually involves a high degree of abstraction and aggregation. On the contrary, microscopic modeling focusses on characteristics and behavior of individual entities. In this case, the aggregate behavior of a system under investigation is not modeled explicitly but evolves from the behavior and interactions of the involved entities [7].

IS ARCHITECTURE A COMPLEX SYSTEM?

We can address architecture in two ways. First, architecture as a ***designed object***, building or an element of the building, and second as a ***design process***.

Design problems in architecture are unique, open-ended and ill-structured [43]. This does not allow architects to rely only on predefined methodologies or previous solutions to similar problems. Even though we apply tested design solutions to new problems, the overall approach in architectural design is typically experimental. Commonly architects take on the responsibility to evaluate all known factors involved in a design project that might lead to unwanted outcomes in both physical (structural, functional) and ethical (social) sense. Architectural problems are complex and characterized by a wide range of determining, partly unknown or subconscious factors on various levels, ranging from local building codes to aesthetic aspects [35].

Moreover, the properties of complex systems lay in the architecture as a materialized object. Even if we take an element of architecture (as an example structural system or a facade) we can consider it as a complex system of building elements working together to create a structure with particular properties and values. Changing the dimensions or any other properties of one element would affect the qualities of the structure as a whole.

CONCLUSION

Arising complexity in architecture forces us to rethink the traditional top-down approach in the benefit of the bottom-up design process [44]. A number of architectural theorists have written about complexity in architecture, among them Jencks, Eisenman, and Lynn, but attempts to use complexity theory in architecture have rarely ventured beyond rather diagrammatic and iconic applications [35].

We can bring an architectural design to a new level by approaching it as a complex system, but in another way then just diagrammatic. Both expressions of architecture as a design process and as a design object can be treated as a complex system. This introduces the first two statements of generative agent-based design:

- ARCHITECTURE IS A COMPLEX SYSTEM

- MODELING AND SIMULATION OF COMPLEX SYSTEMS IS A DESIGN PROCESS

DIGITAL REVOLUTION

"The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it."

MARK WEISER

In the middle of the 20th century, the shift from mechanical and analog electronic technology to digital electronics marked the beginning of the Digital Age. The invention of the transistor led to the development of general-purpose computers, hence the ability to easily move digital information between media [45]. Digital logic circuits, Internet, personal computer and other innovations changed our life. They transformed the industries and created new ones. Digital Era for architecture started with Computer Aided Design (CAD), and Computer Aided Manufacturing (CAM) tools and further developed with Building Information Modeling (BIM) and Generative (or computational) design.

CAM

In 1957 Dr. Patrick J. Hanratty ("Father of CAD/CAM") developed an early commercial numerical control programming language PRONTO, which was the first CAM software system [46]. CAM software is used to control machines for manufacturing and processing of components or designed objects. Today we call this process Digital Fabrication. Generally, computer-aided manufacturing requires a 2D or 3D digital copy of an object, which has to be modeled in CAD software.

CAD

In 1963 Ivan Sutherland developed a first Computer Aided Design system called SKETCHPAD (*Figure 11*). It was more than just a 2D drafting tool, it was integrating the benefits of computation to the design process. The geometry displayed on the screen was a representation, not just of the apparent graphics (points, lines, etc), but also the underlying constraint model, which included the symbolic representations of constraints (anchor points, parallel and orthogonal constraints between lines, etc). The manipulation of one geometric element by the user triggers a new resolution of the constraint model, which in turn propagates changes to the other geometric elements [47]. It took a few decades for CAD/CAM systems to become affordable and feasible to use in practice. Consequently, the software was simplified. It was not a true computer aided Design anymore; it became just a drafting tool or computer-aided Drawing.

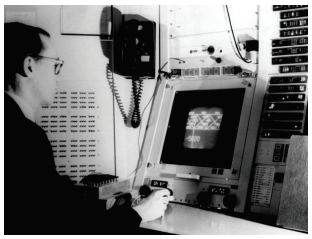


Figure 12:
Ivan Sutherland on MIT Lincoln Labs' TX-2 computer [48]

BIM

In 1987 Graphisoft launched ArchiCAD which regarded as the first implementation of Building Information Modeling. BIM extended the notion of computer as a design tool by adding more dimensions. Three spatial dimensions, time and money as fourth and fifth. BIM is based on the idea of creating a single 3D building model with drawings extracted from the model considered as derived data. The focus of BIM is data management. It is inherently a much more systematic approach to the application of computing to design [47]. As a tool, it is powerful in optimizing the management and logistics but it lacks the freedom of design by limiting the formal possibilities.

GENERATIVE DESIGN

Fundamentally, the task of the computer as a tool is very straightforward: executing a set of instructions and storing the information or data. From that sense, when using BIM or CAD software, a designer has limited access to the data and only a predefined set of instructions. To fully benefit from design with a computer as a tool it is crucial to have control over the instructions or algorithms. This is where generative design comes in. It can also be referred to as parametric or computational design [49]. With computational or generative design, there is a clear distinction between a generative description of the design and generated model. Thus, it reframes the designer's task from the design of a single physical object with a fixed form, to the design of systems, which encode the full range of formal possibilities for a particular design concept [50]. This fundamentally changes the design possibilities and makes it viable to integrate simulation, optimization and digital fabrication in the process [47].

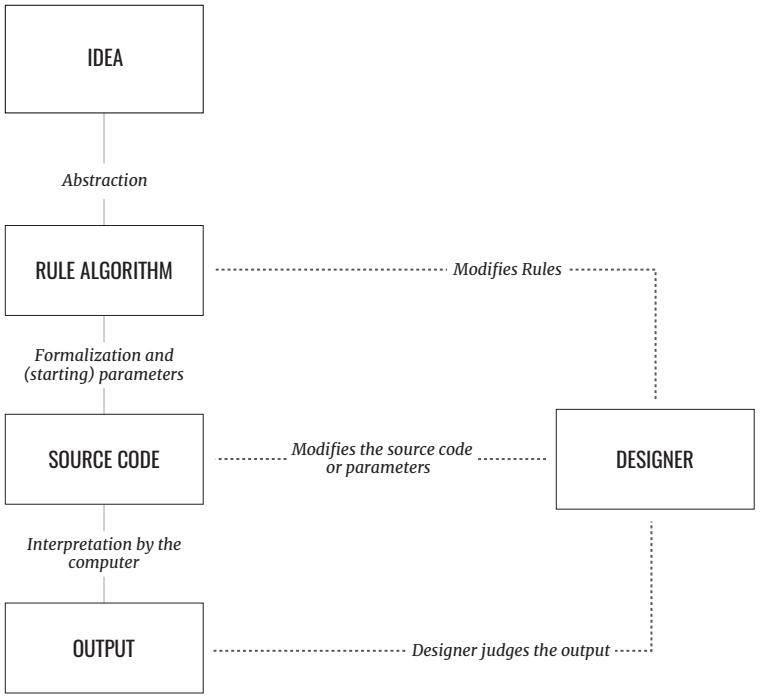


Figure 13:
Generative design process diagram. Redrawn by the author based on [51]

The main challenge in generative design is to create a proper algorithm that generates possible design solutions. It requires a designer to have a specific control strategy for the algorithm. In other words, the way a designer can manipulate the automated process to have a desirable result. There are two distinguishable types of strategies that are commonly used: Direct and Indirect [52].

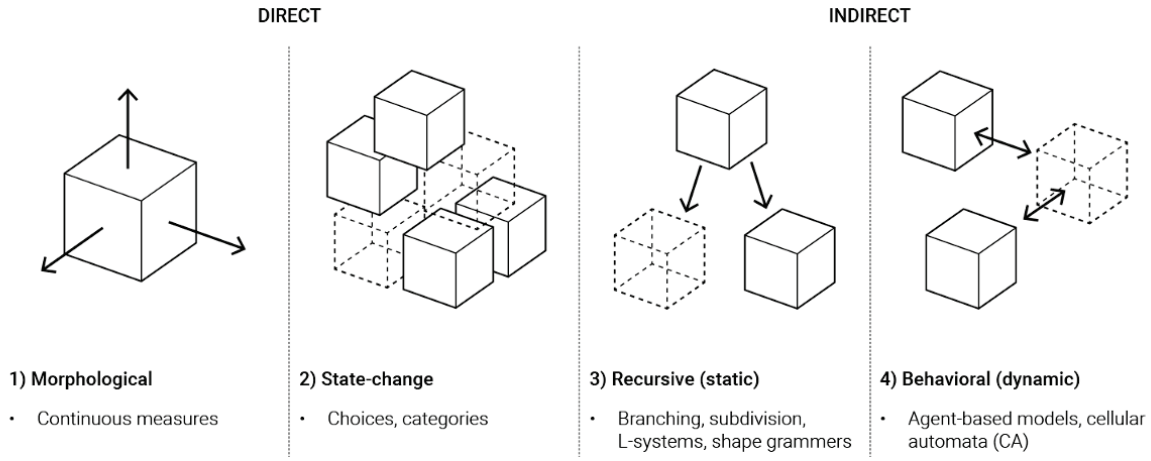


Figure 14: Generative design control strategies [52]

For direct control strategies, it is typical to manipulate the algorithm by changing the input parameters. These parameters could be just numbers or the list of possible choices. This can be very effective in giving direct control to the designed object. However, because direct control strategies can rely on the parameterization of each individual component, they often either do not produce enough variations of design or require too many parameters to effectively generate the designs.

Indirect control strategies include Recursive and Behavioral strategies. Recursive control strategy usually includes one or more recursive algorithmic functions that are controlled through a set of parameters. Behavior strategies control discrete algorithmic agents that interact with each other to create the final design. This strategy is also called agent-based. Indirect strategies are powerful because they describe complex processes that can be controlled using a smaller set of more abstract input parameters [52].

CONCLUSION

Digital revolution produced a set of tools that has changed many aspects of architectural practice as well as the design process. In comparison to other computer tools brought by the Digital Revolution, the biggest advantage of generative design is the ability to create solutions for substantially complex problems that would otherwise be resource-exhaustive or impossible with an alternative approach. This makes it a more attractive option for solving problems with a large or unknown solution set [53].

When looking at the aspects of generative architectural design and complexity theory more closely, the generative approach to architectural design sticks out as a promising strategy [35]. In the design process where architecture is addressed as a complex system, agent-based or generative strategies are the most prominent. If we assume a problem domain, that is particularly complex, large or unpredictable, such as architecture, then it can be reasonably addressed by developing a number of functionally specific and modular components (agents) that are programmed to solve a particular task [30]. Agent-based generative design reflects the statement that the modeling and simulation of complex systems is a design process. It answers the necessity of tool for architectural design as a complex system.

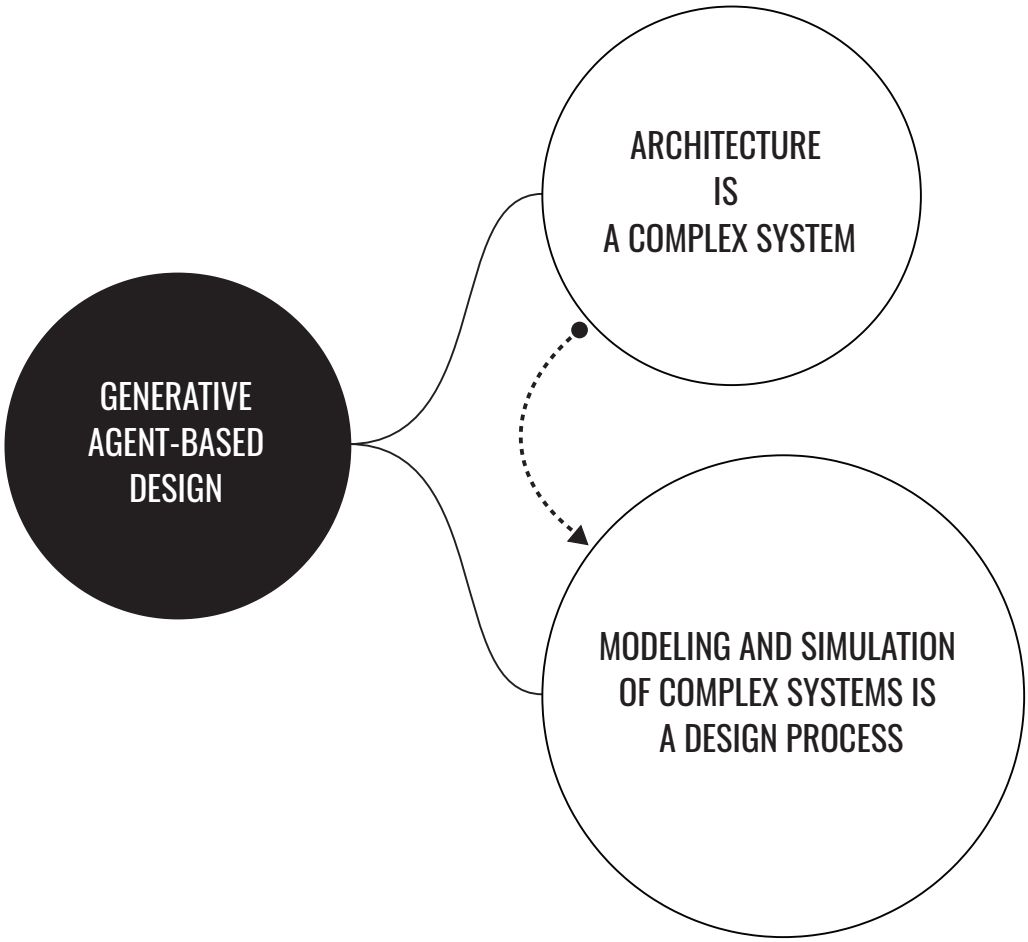


Figure 15:
Diagram of generative agent-based design as a new paradigm.

POTENTIAL OF GAD 2.2.

Generative agent-based design as a technique and methodology for architectural design has several advantages over other generative design techniques. Since the agent-based modeling resembles the modeling and simulation of complex systems, it possesses the qualities of complex systems. We can highlight some advantages and benefits that give the answer to the question where it could lead:

1. EMERGENCE

One of the most distinctive features of agent-based generative design in comparison to other generative design paradigms is that it can produce emergent solutions [7]. Emergence and self-organization arise from the properties of the agent-based system, which is a complex system. Interaction between simple components of a system creates properties or qualities that single distinct element, could not have [54].

An example of emergence is a human consciousness, which is a result of the interaction of 100 billion neurons. Applied to generative design, the interaction of simple agents can produce emergent complex design solutions that are not possible or too difficult to achieve with other methods.

2. SELF-ORGANIZATION

Dynamic interaction of agents in complex system develop in self-organization. Self-organization refers to the ability of a system to change its structure or complexity without any explicit control [55]. In that sort of generative design process, the designed system can adapt according to the environment and constraints to produce solutions that are connected with the design objectives.

3. COMPLEXITY

The complexity of architectural design as a process and as an object arises with time. Generative agent-based design as a method can create solutions to substantially complex problems that would otherwise be resource-exhaustive with an alternative approach. This makes it a more attractive option for problems with a large or unknown solution set [56]. The complexity of agents can be articulated freely. The agent-based systems can also be nested within each other to solve sub-problems.

4. BOTTOM-UP DESIGN

A bottom-up approach is the piecing together of systems to give rise to more complex systems, thus making the original systems sub-systems of the emergent system [57]. A top-down approach is good for understanding something that is already there, and the bottom-up approach is good for constructing and assembling something. This approach is common in natural systems. In bottom-up design, there is no mastermind to give distinct orders for the design solutions. The design is build up by connecting discrete elements into a system that is creating emergent design solutions.

5. INTUITIVE MODELING

Agent-based modeling provides a natural description of the system. The ontological correspondence between the computer agents in the model and real-world actors makes it easy and evident to represent actors, the environment, and their relationships [14].

3. HOW ?

If architecture is a complex system and modeling and simulation of complex systems with the agent-based approach as a design process, ***how*** can we do generative agent-based design? What are the issues and challenges we need to solve when we think about the generative agent-based design as a process? What are the components of this process?

3.1. ISSUES AND CHALLENGES

In current practice of generative agent-based design, several problems stop the approach from being implemented in real life projects and to gain acceptance in computational design field. Issues are highlighted through an analysis of the state of the art projects and developments in architecture.

1. TOOLS FLEXIBILITY, REUSABILITY

In most cases, studied projects are using very specialized and limited tools. For instance in [23] software was built specifically for the purposes of one project, the reusability of the software is questionable. Even though in [24,26] are emphasizing on the importance of flexibility of the tool, authors do not explain the whole framework and building blocks, only introducing some of the used methods to create it.

To be able to create certain behaviors and their variety the instrument for generative ABM must be flexible. There is a necessity to have a common framework and methodology of how to create the agent-based models so that designed agents could communicate with no constraints and created behavioral strategies that can be reused for other projects.

2. LIMITED BEHAVIORAL MECHANISMS

Most of the projects are implementing the swarm logics and using the behavioral rules introduced by Raynolds [29]. The rules are easy to follow and integrate into any agent-based model. However, the simplicity of these behavioral mechanisms emerged in overuse of the swarm logics in all GAD projects. The limitation of the tools available for agent-based modeling is also constraining the designers in creating their own behavioral mechanisms for agent-based generative design.

It is necessary to invest in development and application of other behavioral mechanisms. In a way that they could be reused and modified to fit for various design problems.

3. ABSTRACTION

It makes sense to use generative agent-based design as long as we can transfer that design into reality or build it. It is an existential question for GAD if we can model the constraints of the real world materiality and constructability into the digital design process.

Generative agent-based design today is commonly used on the very first mostly abstracted design phase. As a result geometry of the forms are complex and not thought through from the perspective of material realization. Designers are trying to go around the problem by suggesting 3D printing technologies as a solution. However, 3D printing industry in construction is still not capable to provide cheap, high-quality solutions. For that reason, generative-agent based design is not implemented widely in the practice but mostly explored in some research projects.

Geometry is a communication language between the abstracted design idea and the real design solution. To deal with the problem of abstraction in GAD, it is necessary to highlight the importance of integration of detailing phase in the process as a part of the agent-based model.

Figure 16:
Simplified design process diagram

GAD AS A DESIGN PROCESS 3.2.

To tackle the problems and find solutions, we need to go back to the statement that GAD is a design process for architecture as a complex system and determine the steps and parts of the process.

To solve any design problem the most important is to detect the issue in the first place. The notion of the problem drives the need for the solution and in general the design process. (*Figure 15*) Here we do not focus on the process of determining the problems, but rather on the design process itself with design objectives and issues in mind.

The benefit of generative agent-based design as a design process is the integration of solutions for the multiplicity of different problems in one procedure. So it makes possible to tackle the design objectives in all its complexity and diversity and produce multiple results (*Figure 16*). Potentially, this promises to solve the problem of *"disconnected"* architecture.

The design process usually goes through several levels, starting from the conceptual basic idea of the design, then development process with adding some details and constraints, and finally fine-tuning and detailing of the designed object. Moreover, in between and inside of every level there could be an evaluation and analysis steps. Potentially GAD can be an approach that is able to take in every design level through the time of project development and automatically adopt the design solutions according to constraints and objectives. This would solve the problem of abstracted results in agent-based methods.

The major challenge is to create a kind of framework behind the "black box" of GAD that can be flexible enough. It is necessary to identify the components of this.

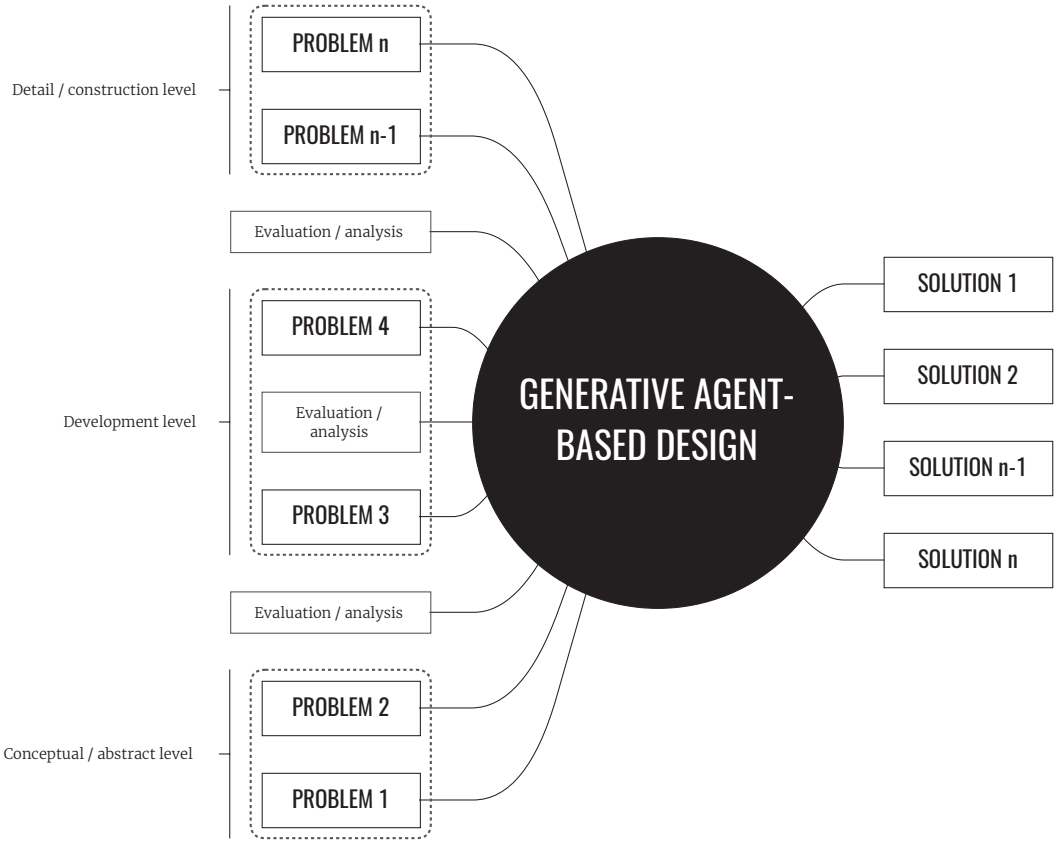


Figure 17:
Diagram of GAD process and levels of design problems

COMPONENTS OF GAD 3.3.

Derived from previous investigations and statements based on complex systems, we can formulate three main components of the generative agent-based design process: design ideas and intentions, agent-based modeling, and simulation of the model (*Figure 17*). These components are the sub-processes, their interconnections create the GAD process.

Design Ideas are the driving force and usually starting point of the process. Ideas should be generated from design problems and objectives. In this part of the process, decisions are made on what exactly is modeled and which approach is used for each problem.

Agent-based modeling is the main part of the GAD. This part is a complex process driven by the design ideas and providing the data for the simulation part. Here it is important to have a modeling framework that is flexible and capable to accumulate any design idea or problem. To solve previously determined issues of GAD the agent-based modeling framework should be suitable for multiple agent behaviors, have building blocks that are easy to reuse in different projects.

Simulation is the process in which agent-based model is carried out. It is the generating process of the GAD, this is where the solutions emerge. Simulation is also connected with design ideas because the simulation can be interactive and the input from a designer can be contributed on the go. Moreover, the results of the simulation could create a feedback loop for the design ideas to emerge.

In this thesis, the focus will be given to the *"Agent-based modeling"* part of the GAD process. The reason for it is that agent-based modeling is usually a binding element that constrains the simulation and design idea components of the process. For *simulation* and *design ideas* components perspective developments will be outlined in conclusion.

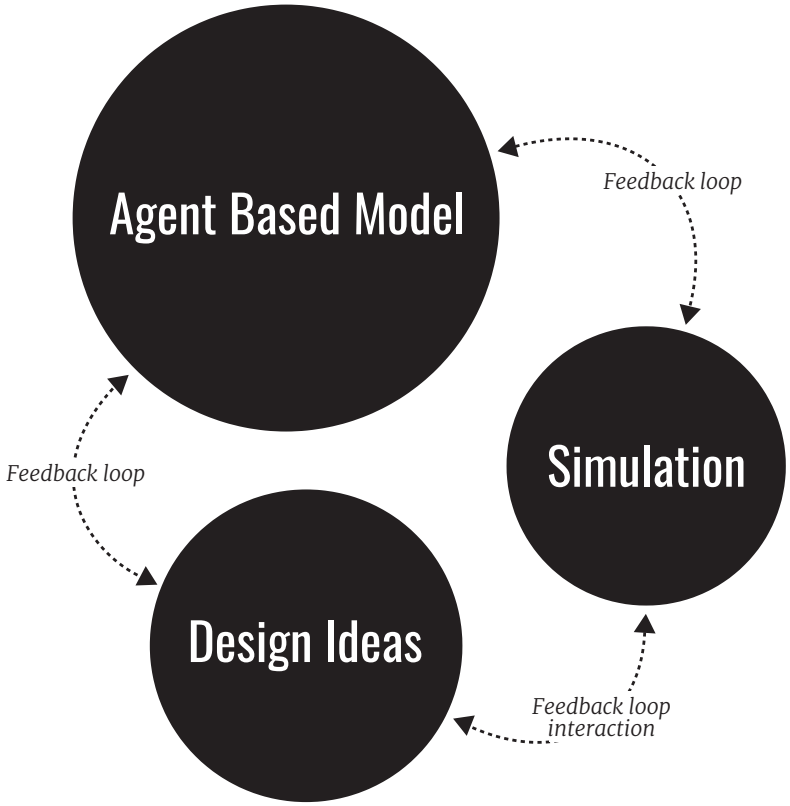


Figure 18:
GAD process components diagram

AGENT-BASED MODEL

The field of agent-based modeling lacks a solid theoretical foundation, efficient approach for model development, and reusability of algorithms. This what General Reference Model for Agent-based Modeling and Simulation is proposed to contribute.

General Reference Model for Agent-based Modeling and Simulation (GRAMS) – is created by Robert Siegfried [7] as the reference model that defines the components and structure of agent-based models as well as constraints for the simulation of such models. It provides a solid foundation of ABM by defining, on a conceptual level, the basic building blocks and their relations.

GRAMS FRAMEWORK

In GRAMS framework agent-based model M is defined as tuple:

M = (T, E, ENT, emb, EV, C)

T - set of points in time

E = (L, P, U) - environment, consisting of a set of locations L, environmental properties P and update functions U

ENT – set of entities

emb: ENT → L – embedding of entities in the environment

EV – event types

C – set of constraints

GRAMS applies to the micro-level as well as to the macro-level of an agent-based model.

MACRO-LEVEL

The macro-level defines the overall model. It contains the common environment and the embedding of all entities currently interacting within this environment. Possible event types and constraints are defined on the macro-level.

Simulation time - **T** is defined as totally ordered set of points in time. It can be bottom-up: defined on micro-level and derived on macro-level. It can be top-bottom: Defined on macro-level and used a subset of it in micro-level.

Environment - **E** denotes the common space in which all agents are acting.

E = (L, P, U): environment consists of **L** locations, environmental properties P, and environmental update functions **U. S(E)** – state of the environment depends on **P, L, U**.

Locations is a concept that helps to embed the entities in the environment. The locations can change in time.

Environmental properties – **P** denotes the properties of the environment in each specific locations. Properties has **D** – domain of properties, which are associated with locations.

Environmental update function – **U** denotes the dynamics of each location's properties. They change the properties P of the environment. Update functions can be triggered by events, and can trigger new events, and can trigger itself.

Embedding of entities - **emb: ENT → L** associates each entity with a specific location in the set of locations **S(L)**. The embedding defines the current position of any entity at any specific point in time, i. e., emb defines the ground truth (i. e., undistorted information) as opposed to perceived truth (i. e., the position assumed by an agent) which may differ.

Events – **e** is defined as an instantaneous occurrence at a specific point in time that may change the state of the model. Event occur instantly and multiple events can take place at the same moment. There are endogenous events and exogenous events. Endogenous events occur and effect only inside of environment or an agent (Micro-Micro, Macro-Macro). Exogenous events occur in one component and effect another component (Macro–Micro). The simulation of the gent-based model can be driven by events or time.

MICRO-LEVEL

The micro-level includes properties of single entities. It contains inner structure and behavior of an agent, and refers to objects without behavior.

Entity set **(ENT)** - denotes all entities, which are part of an agent-based model. It consists of set of agents and set f entities.

An *agent* **(A)** - is defined as tuple **A = (ATT, SEN, EFF)** with ATT representing the set of attributes, SEN the agent's sensors, EFF agent's effectors.

An *object* **(O)** - is an agent without behavior. It is defined by the set of attributes ATT.

Attributes **(ATT)** - are the set of properties and characteristics of agents and objects. Attributes can be static or dynamic. The attributes can be both, simple number values or a complex data structures.

Sensor **(SEN)** provides perception capabilities to an agent. Sensors can be triggered by endogenous and exogenous events. Can trigger new endogenous events. An agent for perceiving its environment primarily uses the sensors. Each sensor may activate effectors of an agent or further sensors. Sensors can be endogenous or exogenous.

Effector **(EFF)** - provides capabilities to actively cause state change. Effectors can be triggered by endogenous events and can trigger endogenous and exogenous events. Effectors can alter the environment, or affect agents embedding in the environment (ex: movement) (exogenous), or change attributes of an agent (endogenous). If effector affects the agent - it is endogenous effector, if it effects the environment - it is exogenous effector.

An *action* refers to a sensor or effector, which was triggered by an event and is currently active.

Sensor-effector-chains are combinations of sensors and effectors that describe the behavior of an agent.

Constraints **(C)** - determine conditions under which successful execution of a sensor or effector action of an agent is possible and therefore represent general laws of the modeled world.

Figure 18 shows the model with all its components in one diagram.

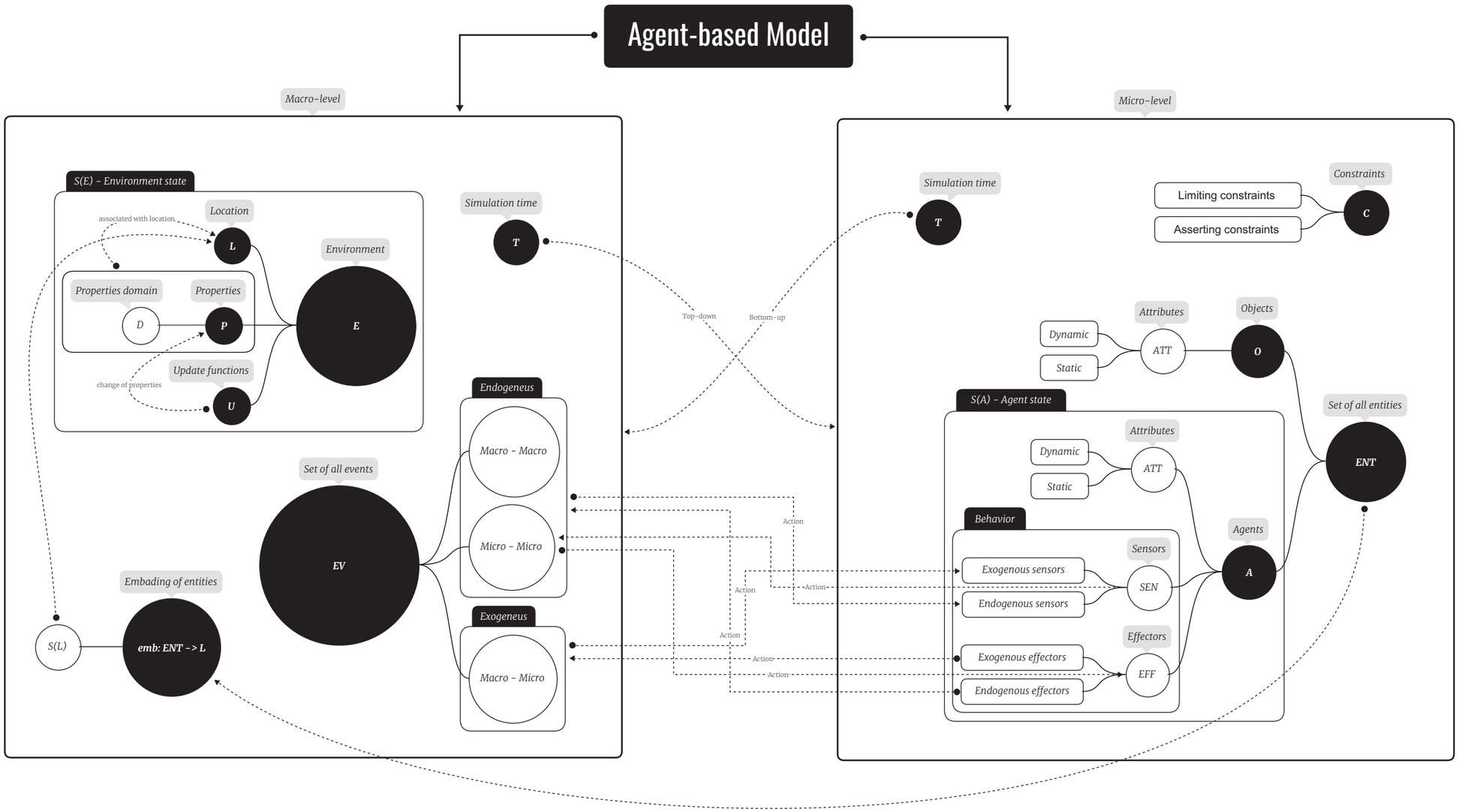


Figure 19:
GRAMS framework diagram

CONCLUSION

> *"First, build your tools"*
>
> *ROBERT AISH*

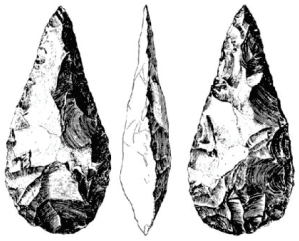


Figure 20:
Drawing of a handaxe from the Acheulean stone tool culture 1,7 million years ago [58]

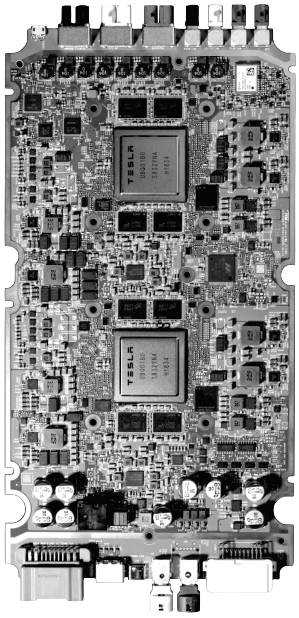


Figure 21:
Full Self-Driving Computer hardware by Tesla - 2019 [59]

Throughout the history the tools where driving and constraining our possibilities. Humanity developed from stone tools (*Figure 20*) to microchips with artificial intelligence (*Figure 21*). However, our challenges are growing in complexity as well. In architecture, digital tools drive new kind of design thinking in architecture to solve current problems. An agent-based generative design is a promising approach that is still in the infancy of its development.

Contribution of this thesis work in pushing forward generative agent-based design are outlined in the answers to the fundamental questions of the field. We formulate a general definition of the GAD, propose the fundamentals of the approach in terms of its objectives, and investigate the process and its components. This research is focused on *"agent-based model"* element of GAD process. It provides the foundation of future development in other elements of the approach.

"Simulation" part requires the development of a digital tools for simulation of the ABMs based on GRAMS framework. In perspective a Grasshopper plugin could be developed, which could give access to new possibilities in GAD.

"Design Ideas" part can be approached from a pragmatic side, by developing an online platform (similar to GitHub) where developers could share the elements of their ABM models. Modular flexibility of the framework allows the reuse of the components. By building up on the previously developed components, it would be possible to address different design problems in an adaptive way.

BIBLIOGRAPHY

1. Apil KC. Issues in Modern Architecture Departure from Past. Asian Architectural Youth Symposium. 2013;
2. Hansmeyer M. From Mesh to Ornament: Subdivision as a generative system. FUTURE CITIES. ETH Zurich; 2010. p. 285–293.
3. Hauck A. What is Generative Design? [Internet]. Linkedin. 2018 [cited 2019 May 31]. Available from: https://www.linkedin.com/pulse/what-generative-design-anthony-hauck/
4. Vitruvius, Rowland ID. Vitruvius: 'ten books on architecture. Howe TN, editor. Cambridge: Cambridge University Press; 1999.
5. Palladio A. Four Books of Architecture. Dover Publications Inc.; 1977.
6. Alexander C, Silverstein M, Ishikawa S. A Pattern Language: Towns, Buildings, Construction (Center for Environmental Structure Series). 1st ed. New York: Oxford University Press, USA; 1977.
7. Siegfried R. Modeling and simulation of complex systems. Wiesbaden: Springer Fachmedien Wiesbaden; 2014.
8. Multi-agent system - Wikipedia [Internet]. [cited 2019 Jun 7]. Available from: https://en.wikipedia.org/wiki/Multi-agent_system
9. Agent-based model - Wikipedia [Internet]. Wikipedia. [cited 2019 May 27]. Available from: https://en.wikipedia.org/wiki/Agent-based_model
10. Poggi A, Tomaiuolo M. Rule Engines and Agent-Based Systems. Encyclopedia of artificial intelligence. IGI Global; 2008.
11. Pantazis E, Gerber D. A framework for generating and evaluating façade designs using a multi-agent system approach. International Journal of Architectural Computing. 2018 Dec;16(4):248–270.

12. Tumblr - aYearInCode(); [Internet]. [cited 2019 Jun 16]. Available from:

https://ayearincode.tumblr.com/post/107414487116/this-morning-i-added-some-new-rules-to/embed

13. Flock (birds) - Wikipedia [Internet]. [cited 2019 Jun 7]. Available from:

https://en.wikipedia.org/wiki/Flock_(birds)

14. Chen L. Agent-based modeling in urban and architectural research: A brief literature review. Frontiers of Architectural Research. 2012 Jun;1(2):166–177.

15. Namazi-Rad M-R, Padgham L, Perez P, Nagel K, Bazzan A, editors. Agent based modelling of urban systems. Cham: Springer International Publishing; 2017.

16. H. van Dam K, Koering D, Bustos-Turu G, Jones H. Agent-based simulation as an urban design tool: Iterative evaluation of a smart city masterplan. London, UK; 2014.

17. Mgr Art Peter Buš. Emergence in Urban Environments: Agent-based Simulation of Environment Reconfiguration. Unpublished. 2016;

18. Sevil Y, J. Gerber D. Prototyping Generative Architecture - Experiments on Multi-Agent Systems, Environmental Performance and 3D Printing. 8th ASCAAD Conference. London (United Kingdom): Imperial House Publishers; 2016. p.145–154.

19. Pantazis E, Gerber D. Designing with complexity, Interactive form finding of reciprocal frames through a multi-agent system. Oulu, Finland: eCAADe (Education and Research in Computer Aided Architectural Design in Eu- rope) and Oulu School of Architecture, University of Oulu; 2016.

20. Gerber D, Pantazis E, Wang A. Interactive Design of Shell Structures Using Multi Agent Systems Design Exploration of Reciprocal Frames Based on Environmental and Structural Performance. Future Trajectories of Computation in Design. Istanbul Technical University; 2017.

21. Sina M, Soungmin Y, Nimish M B. Multi-scalar agent-based complex design systems- the case of CECO (climatic-ecologies) studio, informed generative design systems and performance-driven design workflows . Design Agency. Los Angeles: Riverside Architectural Press; 2014. p.111–116.

22. Tsiliakos, Marios. Swarm Materiality: A multi-agent approach to stress driven material organization. In Digital Physicality. Czech Technical University in Prague; 2012.

23. Scheurer F. Turning the Design Process Downside-up. In: Martens B, Brown A, editors. Computer aided architectural design futures 2005. Berlin/Heidelberg: Springer-Verlag; 2005. p. 269–278.

24. Baharlou E, Menges A. Generative agent-based design computation: Integrating material formation and construction constraints. 2013. p. 165–174.

25. Groenewolt A, Schwinn T, Nguyen L, Menges A. An interactive agent-based framework for materialization-informed architectural design. Swarm Intell. 2017 Dec 4;12(2):1–32.

26. Baharlou E, Menges A. Toward a Behavioral Design System: An Agent-Based Approach for Polygonal Surfaces Structures. 35th Annual Conference of the Association for Computer Aided Design in Architecture (ACADIA). Cincinnati: Atropos Press; 2015.

27. Guo Z, Li B. Evolutionary approach for spatial architecture layout design enhanced by an agent-based topology finding system. Frontiers of Architectural Research. 2017 Mar;6(1):53–62.

28. Holland JH. Emergence: From Chaos To Order. Oxford Univ Pr; 2000.

29. Reynolds C. Steering Behaviors For Autonomous Characters. Foster City, California: Sony Computer Entertainment America; 1999. p. 763–782.

30. Gerber DJ, Pantazis E, Marcolino LS. Design Agency. In: Celani G, Sperling DM, Franco JMS, editors. Computer-Aided Architectural Design Futures The Next City - New Technologies and the Future of the Built Environment. Berlin, Heidelberg: Springer Berlin Heidelberg; 2015. p. 213–235.

31. Reas C, Fry B. Processing: A Programming Handbook For Visual Designers And Artists. The Mit Press; 2007.

32. Sugihara S. iGeo: Algorithm Development Environment for Computational Design Coders with Integration of NURBS Geometry Modeling and Agent Based Modeling. Design Agency. Los Angeles: ACADIA; 2014.

33. STUDIO ROLAND SNOOKS - composite wing [Internet]. [cited 2019 Jun 17]. Available from:

http://www.rolandsnooks.com/compositewing

34. Markussen T. Fact or Fuzzy? On the Topology of Space Perception in Digital Architecture. 2008.

35. Herr CM. Generative Architectural Design and Complexity Theory. 2002.

36. Paradigm - Wikipedia [Internet]. [cited 2019 Jun 4]. Available from: *https://en.wikipedia.org/wiki/Paradigm*

37. Gleick J. Chaos: Making A New Science. Minerva; 1997.

38. Castellani, Brian (2018) "Map of Complexity Science" Art & Science Factory. [Internet]. [cited 2019 Jun 2]. Available from: *https://www.art-sciencefactory.com/complexity-map_feb09.html*

39. An Introduction to Complexity Theory – Jun Park – Medium [Internet]. [cited 2019 Jun 1]. Available from:

https://medium.com/@junpo1/an-introduction-to-complexity-theory-3c20695725f8

40. Boccara N. Modeling Complex Systems. New York, NY: Springer New York; 2010.

41. Gordon D. Ants at Work : How an Insect Society Is Organized. Free Press; 1999.

42. Scientific modelling - Wikipedia [Internet]. [cited 2019 Jun 3]. Available from:

https://en.wikipedia.org/wiki/Scientific_modelling

43. Rittel HWJ, Webber MM. Dilemmas in a general theory of planning. Policy Sci. 1973 Jun;4(2):155–169.

44. de Almeida CRP, Pratschke A, Rocca RL. In-Between and Through: Architecture and Complexity. International Journal of Architectural Computing. 2005 Sep;3(3):335–354.

45. Digital Revolution - Wikipedia [Internet]. [cited 2019 Jun 4]. Available from:

https://en.wikipedia.org/wiki/Digital_Revolution

46. Aouad G, Wu S, Lee A, Onyenobi T. Computer Aided Design Guide for Architecture, Engineering and Construction. illustrated. Routledge; 2013.

47. Peters B, Peters T. Inside smartgeometry: expanding the architectural possibilities of computational design. Chichester, West Sussex, United Kingdom: John Wiley & Sons Ltd; 2013.

48. Ivan Sutherland demonstrating Sketchpad on the TX-2 - CHM Revolution [Internet]. [cited 2019 Jun 4]. Available from: *https://www.computerhistory.org/revolution/input-output/14/349/1878*

49. Nagy D. Introduction to computational design – Generative Design – Medium [Internet]. medium. 2017 [cited 2019 May 23]. Available from: *https://medium.com/generative-design/introduction-to-computational-design-6cofdfb3f1*

50. The design space – Generative Design – Medium [Internet]. [cited 2019 Jun 4]. Available from:

    https://medium.com/generative-design/step-1-generate-6bf73fb3a004

51. Bohnacker H, GroÃŸ B, Laub J, Lazzeroni C. Generative Gestaltung. Schmidt Hermann Verlag 2009-11-01; 2009.

52. Control strategies – Generative Design – Medium [Internet]. [cited 2019 Jun 4]. Available from:

    https://medium.com/generative-design/control-strategies-b4cfo7b26cda

53. Prasanta B. Compositional model-based design: A generative approach to the conceptual design of physical systems [Doctoral dissertation]. 1993.

54. Emergence - Wikipedia [Internet]. Wikipedia. [cited 2019 May 31]. Available from:

    https://en.wikipedia.org/wiki/Emergence

55. Krolikowski R, Kopys M, Jedruch W. Self-Organization in Multi-Agent Systems Based on Examples of Modeling Economic Relationships between Agents. Front Robot AI. 2016 Jul 29;3.

56. Generative design - Wikipedia [Internet]. Wikipedia. [cited 2019 May 31]. Available from:

    https://en.wikipedia.org/wiki/Generative_design#cite_note-8

57. Top-down and bottom-up design - Wikipedia [Internet]. [cited 2019 Jun 5]. Available from:

    https://en.wikipedia.org/wiki/Top-down_and_bottom-up_design

58. File:Bifaz lanceolado-San Isidro (Madrid).png - Wikimedia Commons [Internet]. [cited 2019 Jun 17]. Available from:

    https://commons.wikimedia.org/wiki/File:Bifaz_lanceolado-San_Isidro_(Madrid).png

59. Tesla. Full Self-Driving Computer hardware by Tesla [Internet]. Twitter. 2019 [cited 2019 Jun 13]. Available from:

    https://twitter.com/Tesla/status/1120480117338398720?s=20