



HAL
open science

Computational Tools in Architecture and Their Genesis: The Development of Agent-based Models in Spatial Design

Nadja Gaudillière

► **To cite this version:**

Nadja Gaudillière. Computational Tools in Architecture and Their Genesis: The Development of Agent-based Models in Spatial Design. RE: Anthropocene, Design in the Age of Humans - 25th CAADRIA Conference, D. Holzer, W. Nakapan, A. Globa, I. Koh, Aug 2020, Bangkok, Thailand. pp.497-506. hal-04588596

HAL Id: hal-04588596

<https://hal.science/hal-04588596>

Submitted on 27 May 2024

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

COMPUTATIONAL TOOLS IN ARCHITECTURE AND THEIR GENESIS: THE DEVELOPMENT OF AGENT-BASED MODELS IN SPATIAL DESIGN

NADJA GAUDILLIERE

¹*Laboratoire GSA, Ecole Nationale Supérieure d'Architecture
Paris-Malaquais, 75006 Paris, France*

¹*nadja.gaudilliere@gmail.com*

Abstract. Based on the assumption that socio-technical networks of computation in architecture exist and must be analyzed deeper in order to understand the impact of algorithmic tools on the design process, the present paper offers a foray into it, drawing on science studies methodologies. The research explores in what regard multi-agent systems (MAS) are representative as much from the existence of these socio-technical networks as of how their development influences the tension between tacit and explicit knowledge at play in procedural design processes and of the strategies architectural designers develop to resolve this tension. A methodology of analysis of these phenomena is provided as well as results of the application of this method to MAS, leading to a better understanding of their development and impact in CAAD in the past two decades. Tactics of resolution shaped by early MAS users enable, through a double appropriation, a skillful implementation of architectural practice. Furthermore, their approach partially circumvents the establishment of technical biases tied to this algorithmic typology, at the cost of a lesser massive democratization of the algorithmic tools developed in relation to it.

Keywords. Computational tools; multi-agent system; architectural practice; tacit knowledge; digital heritage.

1. Introduction

The Digital Turn in architecture is a widely accepted notion since the famous AD special issue *The Digital Turn 1992-2012*, edited by Mario Carpo and stating the existence of experimentations around the idea of the digital in architecture as well as popularizing the term (Carpo 2012). The Digital Turn as currently defined encompasses the use of a variety of digital tools. The computational movements in architecture are nevertheless also characterized by a specific socio-historical context of emergence (Gaudillière 2019). Most of its architectural production consists in paper projects, prototypes, pavilions, produced in an academic environment. The computational movements therefore organise around a series of practitioners, of research units and institutions, shaping a network of knowledge transmission. This academic set-up has enabled the field to blossom

away from most of the usual constraints of the construction industry. This freedom and diversity of explorations make its production a prefiguration of the global computational turn in architectural design currently happening. The computational field as it has developed in the last 50 years forms therefore an ideal set of designers and projects to study programming-based spatial design and the translation of architectural constraints into computational design tools. Furthermore, as computational movements in architecture are characterized as much by the technical aspects of the practice as by the structure of its community of practitioners, they can be identified as a socio-technical network (STN) (Latour 1991), still to be described as such.

The present researches focuses on the analysis of STN of computation in architecture for a specific algorithmic typology, multi-agent systems (MAS), and studies the trade-off happening between the mobilization of tacit knowledge and the explicitation of formal instructions when resorting to it for architectural design purposes, as well as how the structure of programming interfaces of MAS has influenced this trade-off. The study highlights three major dynamics influencing the development of MAS in CAAD: appropriation, democratization, rationalization. The paper will first address the characteristics of the algorithmic typology studied, before presenting the methodology on which the research is based and discussing the findings regarding the three dynamics identified, analyzing to what extent the resort to algorithmic design tools impacts architectural practice.

2. Multi-Agent Systems as a typology

MAS, also referred to as swarms, are object-oriented algorithmic systems modeling the behavior and interactions of sets of agents. Each agent is obeying a series of given laws, and their interactions generate what is known as emergent behavior: a conduct of the set that differs from the specific behavior instructed to each agent. The components of a MAS are the following : an *environment* - usually a measurable space -, *objects* populating this environment - that can be modified by the *agents, relations* tying objects and agents together, and *operations* enabling the interactions between objects and agents (Ferber 1995). Amongst the early developments of MAS are the Boids, by Craig Reynolds, in 1986. Craig Reynolds, a computer scientist and graphic designer working at the time on the animation of large numbers of elements for the movie industry, programmed the Boids to simulate the behavior of a bird flock. While being one of the first MAS developed, the flocking algorithm devised by Reynolds also constitutes an example of the type of rules that are applied to agents in swarm algorithms. The Boids obey three rules: separation - respecting a minimal distance from other agents - alignment - head towards the average displacement direction of the flock - and cohesion - respecting a maximal distance from other agents (Reynolds 1987). Following this breakthrough, a large number of frameworks were developed to enable programming with MAS from 1995 on, and those have since been regularly used, mainly in the film and video games industries. In particular, the pioneering development of MASSIVE in 1996 for the animation of crowds of thousands or more in the *Lord of the Rings* films, and the Golaem Crowd Maya plugin,

developed in 2011, are nowadays still in use in the industry.

Since MAS can be identified as a specific typology of algorithm, they imply the resort to a predefined typology of instructions, identical for all algorithms (figure 1). The MAS typology as used in CAAD entails three levels of instructions, in the form of rules. First, the rules pertaining to the initial state of the system, such as the position or entry point of the agents and the position of the objects. Secondly, behavior rules for the agents are implemented regarding their movement - direction rules, such as the boids' separation, alignment and cohesion - and their interactions with the objects - attraction or repulsion for example. Finally, rules relating to the geometrical exploitation of the MAS output, in order to convert it in data relevant to the architectural object conceived, by using the final position of the agents, the traces of their displacement or of the operations, and potentially by adding novel geometrical elements.

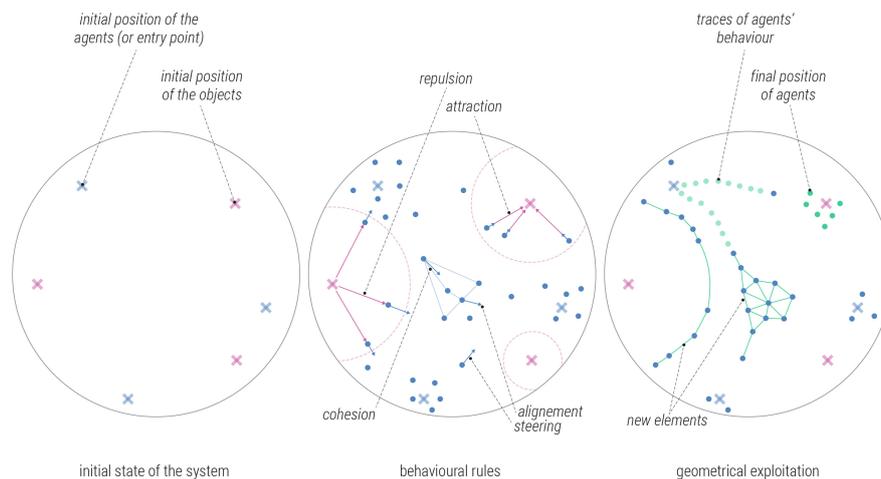


Figure 1. Rule typology for multi-agent systems.

3. Methodology

The research is based on the analysis of a series of case studies by practitioners key to the development of the MAS typology, its related tools and its use for architectural design. In order to outline the computational field as described in the introduction, AD issues have been searched to establish a list of projects and practitioners resorting to digital tools, with a focus on the use of scripting to produce architectural objects or information relevant to the production of an architectural object. The compiled list highlights practices, institutions, projects and tools of the computational movements in architecture, and identifies each project with one or more typologies of algorithms. From this list, the projects associated with MAS were extracted and practices and practitioners appearing

more than three times in the list were identified as having made a significant contribution to the development of their use in CAAD. In complement, a literature review of published projects resorting to MAS in the Cumincad database was conducted on 115 papers to assess the contemporary use of MAS, after the creation of a number of algorithmic tools enabling an easier manipulation. The chronological development of major algorithmic tools for MAS modelling has also been examined, for standard frameworks as well as the specific tools involved in the case studies - such as Toxiclibs for Processing and the Flower Power Rhino plugin - as well as for grasshopper plugins - Quelea, Zebra, Nursery, Boid, Physarealm, Culebra - offering an overview of the democratization of this typology through tool-making.

Three aspects are at stake in this study: the constitution of the STN and the chronologic development of MAS use in the computational movement in architecture, the development of ready-made tools and the trade-off between tacit knowledge and programming instructions formalisation. A grid of analysis for the projects themselves has been developed, and each case study has been analyzed following this same grid of criteria, containing four parts. First, the algorithmic specifications and the choices made for the MAS rules are examined. The articulation with other typologies of algorithms is also assessed, in the cases where the MAS is only a part of the complete architectural design algorithm. Secondly, the architectural instantiation of the algorithm is examined. The goals targeted while using the swarm are sorted in four categories: generating an initial state, a space hierarchisation, structural elements and shapes. Architectural specifications such as the ones originating in the brief or the context and their implementation are also recorded. Finally, information pertaining to the technical set-up - type of programming interface, visualization software and programming software - and the organization chart - composition of team, role, background, programming skills - are collected. The sources of information used in the analysis are interviews with practitioners, public documents such as finalized drawings for the projects - including diagrams explaining the algorithmic structure - and documents found in public and private archives, in particular programming scripts, but also complementary sketches and documents throughout the genesis of the project.

4. Multi-Agent Systems as tailored algorithms : tacit-explicit tension and appropriation

As MAS blossomed in the 1990s, architects started resorting to them as well (Krause 1997, Coates & Schmid 1999), and these algorithms gained further momentum in CAAD during the 2000s. The emergent behavior of swarms and their ability to self-organise turned them into popular devices among architects of the computational field, as well as their capacity to handle complexity and the possibility they encapsulate to program matter (Pantic & Hahm, 2015). Users devised various applications based on swarms behaving according to diverse rules, from simple flock algorithms to simulations of magnetic fields (Andrasek 2009) or behavioral data (Schumacher 2018). More complex uses were also developed, such as algorithms combining several swarms together, or with other algorithmic

typologies. A few examples of the resort to MAS for architectural design can be given. The project Mesonic Fabrics, by biothing, associates a MAS with cellular automata in order to generate roofing structures. The Cliff House project, by kokkugia, is an experimentation tying MAS to the use of composite fiber architecture to develop a structural shell for a house on an extreme cliff topology. A third example is the Trabeculae / Protosynthesis project by supermanoeuvre, reimagining an office tower and its atrium by implementing two MAS, one shaped by light transmission requirements and the other creating a structural truss network. Finally, Living Morphologies (supermanoeuvre) revisits Le Corbusier's Unité d'Habitation with a MAS retro-engineering the circulation logic devised by the architect. These explorations of the potential of MAS for CAAD were accompanied by the development of tailored algorithms, as well as libraries developed by emblematic practices of the field, in particular supermanoeuvre, Kokkugia and biothing. This first period of appropriation of MAS by architects is characterized by the exploration of multiple architectural issues - structure, light, form, users' movements -, and a small-scale STN, depicted in figure 2. This phase is also marked by the high programming skills of MAS users, resulting in a skillful negotiation of the tacit-explicit tension.

Practitioners generally draw on a form of subjectiveness and on their expertise to produce a relevant answer, in the form of an architectural object, to a given spatio-temporal context. The expertise built by architects throughout their years of training and practice relies on intuition as well as on tacit knowledge. The latter is a form of knowledge that cannot be formalised or rendered explicit by the person holding it and is therefore hardly transferable (Collins 2010). It cannot be put in writing neither given the shape of a set of instructions. Whereas tacit knowledge is instrumental to the practice of a discipline, given its inability to be formalised, it also renders specific disciplines very hard to automate and specific sets of knowledge very hard to transfer to computer-supported programs. This points out a key issue for architecture: it is first and foremost a practice, and should be analyzed as such, including in the understanding we have of how architects resort to algorithmic design tools. The fact that a computer relies on sets of explicit knowledge, in the form of instructions transmitted through programming languages, to execute series of calculations, not only makes it difficult to automate some practices, it also requires from architects using algorithmic tools to embrace a procedural understanding of the design process. This renewed understanding of the design process is nevertheless at the core of the extensive potential seen in algorithmic design tools and of the renewal and enrichment of architectural production in the last decades. While the superimposition of the practical dimension of architecture and the formal dimension of computation creates a negotiation between tacit and explicit, practitioners of the computational movements have since their dawn developed strategies to tackle it. The core STN associated with the development of MAS is the illustration of one strategy in particular, resulting in the appropriation of the typology.

(Gaudillière 2020). This can be the result of a resort to complex algorithmic typologies originating in other disciplines - such as biology for evolutionary tools - with a rapid democratization and the lack of an appropriation period through the development of tailored algorithmic design tools by and for users in the field of architecture. In the case of MAS, however, the practitioners of this first phase represent precisely this appropriation period, and despite the typology originating in other disciplines, algorithmic tools currently in use have been mostly devised by the practitioners themselves, resulting in the overlapping of developers and users competences.

The programming skills observed also ensue in a strong bias awareness. The combination of programming mastery and bias awareness results in a fully grasped negotiation between tacit knowledge and explicit instructions. Both the interviews and the analysis of the scripts attest to the mastery and understanding of technical set-ups and typological rules, enabling a conscient mobilization both of tacit knowledge and of these technical aspects in architectural implementation. Nevertheless, while the mobilization of MAS through both the genesis of libraries and the scripting of tailored algorithms relying on these libraries for each project confirms the appropriation of this algorithmic typology, the segmentation of project specific algorithms hints to a delineation of the process. The early version of the Flower Power plugin, relying on the previous and separated drawing of the plan, is an example. Another example is the description of the kokkugia libraries and their two-steps use in the practice : first devising MAS behavior libraries by expliciting instructions and secondly, choosing the library based on known - formal - results of the behavior and on a intuition regarding a specific project, to build a tailored script based on it. The practitioners observed thus recreate a classic mobilization of tacit knowledge, therefore demonstrating an appropriation of this algorithmic typology for architectural design.

5. Multi-Agent Systems as ready-made tools : interfaces for democratization

More recently, a second phase of MAS development in spatial design has begun, as this typology, alongside many algorithmic design tools, has initiated a massive democratization, with the appearance of numerous new tools designed for an easy manipulation of the typology (figure 3). The resort to MAS currently addresses three major fields: behavioral simulation for urban planning, self-organizing workflows and decision steering, and complex shape generation for architectural objects. While the first phase is characterized by one major category of users, the second phase, given its democratization dynamic, shows new types of users, with distinctive ways of mobilizing this algorithmic typology. The six Grasshopper plugins allowing for the use of swarms show a cumulated number of 56 636 downloads, an average of 8090 per plugin - Kangaroo Physics, the most downloaded Grasshopper plugin, displays a total of 443 415 downloads. While this already hints to a small community of users, active members of corresponding Grasshopper groups only represent less than 2% of this number, as well as published papers on the topic. Despite the difficulty to assess properly the number of regular users of such algorithmic typologies, these figures outline a key element regarding the field of MAS for CAAD in academia: the differentiation

between superficial and extensive uses. These elements suggest that, while the presence of Grasshopper plugins hints to attempts at democratising MAS through easier interfaces, the pool of both types of users remains small, in particular when compared to other typologies such as evolutionary tools, currently much more widespread (Gaudillière 2020).

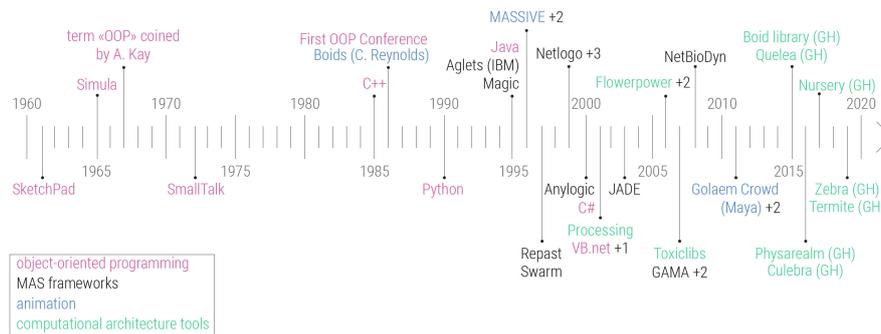


Figure 3. Chronology of multi-agent systems programming principles, frameworks, and tools.

To understand the mechanism of democratization for algorithmic design tools in architecture, a key element to study is the interface of tools, and its thickness. The notion of thickness of the interface acknowledges the varying distance existing between the initial architectural intention and the final representation that is the output of the algorithm, and is similar to the multiples *layers* of an interface that UX designers refer to (Masure 2014). The variation of thickness of the interface is intricately connected to the ability of the user to access rules and details of the mathematical model on which the algorithm is based and to the similarity of knowledge profile of users and developers. Therefore, in the first phase, as users are in their majority capable of scripting a tailored algorithm and to devise their own libraries and interfaces, the knowledge profile overlaps, as previously highlighted, thus providing almost zero-thickness interfaces of MAS. Interfaces and the layers forming them for both tailored algorithms and ready-made algorithmic tools, each typical from one phase, are depicted in figure 4. While tailored algorithms enable an access to rules in upper layers, and are composed of a small number of layers, ready-made tools are composed of many layers and the access to the programming of specific rules is located in lower layers, harder to access for users with little programming skills and leaving upper layers with only predetermined behaviors. Thus, the second phase of development of MAS is characterized by interfaces with high thickness, hardly enabling a detailed manipulation of the typology and hindering appropriation through tacit implementation, a step crucial to the use of MAS given the complexity of this typology, and therefore to their democratization.

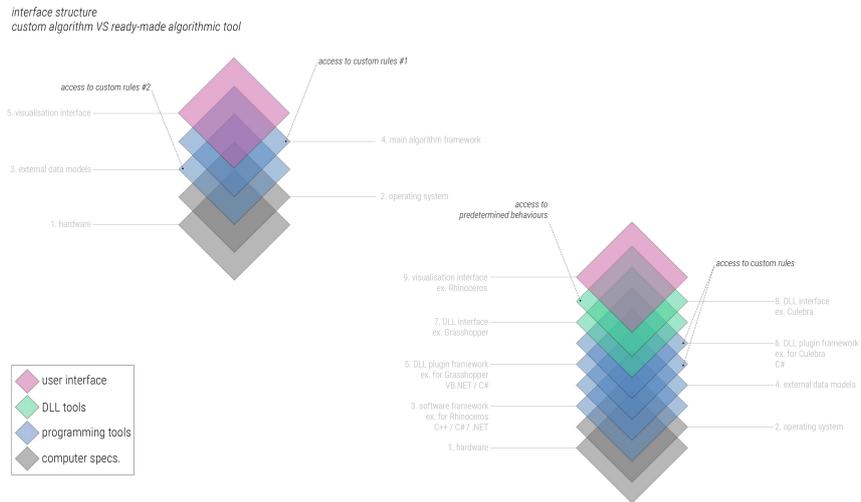


Figure 4. Interface layers for ready-made and tailored algorithms.

Of aforementioned strategies of negotiation between tacit and explicit depends the becoming or not black boxes of algorithmic design tools. By providing an easier resort to algorithmic design tools, interfaces display a black box mechanism (Latour 1987) : they become standard tools of architectural design, and users do not need any longer to question in detail their functioning and relevance, no need to scratch beyond the interface to fully grasp the complex mathematical, physical and informational models they are based on. While the simplification through an interface easier to manipulate is key to the democratization of algorithmic design tools in architecture, tactics of negotiation depend in a large part of the understanding of tools by their users, and on biases enabled by the structuration of algorithmic typologies, of tools and of interfaces.

6. Conclusion

Based on the assumption that socio-technical networks of computation in architecture exist and must be analyzed deeper in order to understand the impact of algorithmic tools on the design process, the present paper offers a foray into it, drawing on science studies methodologies. The research explores in what regard MAS are representative as much from the existence of these STN as from how their development influences the negotiation between tacit and explicit knowledge at play in procedural design processes. A methodology of analysis of these phenomena is provided as well as results of the application of this method to MAS, leading to a better understanding of their development and impact in CAAD in the past two decades. Tactics of negotiation shaped by early MAS users enable a double appropriation - borrowing an algorithmic typology from other fields to create tailored tools from it and managing to implement a classical architectural

practice by mobilizing tacit knowledge. Furthermore, their approach partially circumvents the establishment of technical biases tied to this algorithmic typology, at the cost of a lesser massive democratisation of the algorithmic tools developed in relation to this typology, displaying especially thick interfaces. MAS are an algorithmic typology that necessitates technical mastery, but in return pushes a sensible use of computation in architectural design, as well as a diminution of epistemological and technical biases. These characteristics clash with the global contemporary trends of architects pulling away from technical issues (Picon 1989, Carpo 2011) and of computation tools as vectors of rationalization of architectural practice for the industry (Gaudillière 2020), further explaining the difficult democratization of this typology, despite what it has to offer.

References

- “Kokkugia” : 2004. Available from <<https://kokkugia.com>> (accessed 21/01/2019).
- “supermanoeuvre” : 2007. Available from <<https://www.supermanoeuvre.com>> (accessed 21/01/2019).
- “Plethora Project” : 2011. Available from <<https://www.plethora-project.com>> (accessed 21/01/2019).
- “Food4Rhino” : 2019. Available from <<https://www.food4rhino.com>> (accessed 12/12/2019).
- Andrasek, A.: 2009, *biothing*, HYG.
- Carpo, M.: 2011, *The Alphabet and the Algorithm*, The MIT Press.
- M. Carpo (ed.): 2012, *The Digital Turn in Architecture 1992-2012*, Wiley & Sons.
- Coates, P. and Schmid, C.: 1999, Agent Based Modelling, *Architectural Computing from Turing to 2000 : Proceedings of the 1999 eCAADe*, 652-661.
- Collins, H.: 2010, *Tacit and Explicit Knowledge*, University of Chicago Press.
- Ferber, J.: 1995, *Les systèmes multi-agents*, InterEditions.
- Gaudillière, N.: 2019, Towards an History of Computational Tools in Automated Architectural Design - The Seroussi Pavilion Competition as a Case Study, *Intelligent & Informed - Proceedings of the 24th CAADRIA Conference - Volume 2*, 581-590.
- Gaudillière, N.: 2020, Biomimétisme & Optimisation : algorithmes génétiques, influence de l’interface & biais techniques dans la conception architecturale assistée par ordinateur, *Projeter l’architecture, aux carrefours du numérique et du vivant*, ENSAPVS.
- Krause, J.: 1997, Agent Generated Architecture, *Design and Representation : Proceedings of the 1997 Association for Computer Aided Design in Architecture (ACADIA)*, 63-70.
- Latour, B.: 1989, *La science en action*, La Découverte.
- Latour, B.: 1991, *Nous n’avons jamais été modernes*, La Découverte.
- Masure, A.: 2014, *Le design des programmes : des façons de faire du numérique*, Ph.D. Thesis, Université Paris-1 Panthéon Sorbonne.
- Pantic, I. and Hahm, S.: 2015, Isomorphic Agency, *Emerging Experience in Past, Present and Future of Digital Architecture, Proceedings of the 20th International Conference of the Association for Computer-Aided Architectural Design Research in Asia CAADRIA*, 178-188.
- Picon, A.: 1989, Les ingénieurs et la mathématisation. L’exemple du génie civil et de la construction, *Revue d’histoire des sciences*, 42(1-2), 155-172.
- Reynolds, C.: 1987, Flocks, herds and schools: a distributed behavioral model, *SIGGRAPH ’87 Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, 25-34.
- Schumacher, P. 2018, Freedom via Soft Order : Architecture as A Foil for Social Self-Organisation, in O. Hopkins (ed.), *AD Special Issue - Architecture and Freedom : Searching for Agency in a Changing World*, Wiley & Sons.