SEMINAR URBANISM

Sueters Bart

Oswald Devisch

design

GAME

>> SCRIPT /E


AAdrl, London

School: Provinciale Hogeschool Limburg
Seminar: Urbanism
Thesis: Design Game
Student: Sueters Bart
Promotor: Oswald Devisch
Course: Master Architecture
Academic year: 2007-2008

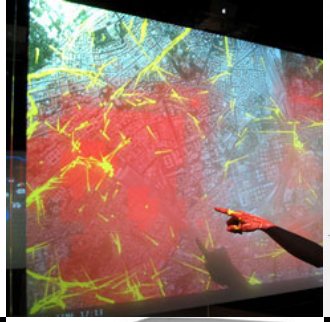
2007-2008
Realisation: Sueters Bart
In collaboration with: Oswald Devisch

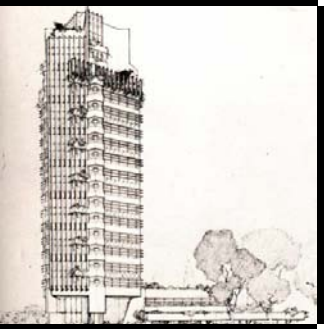
Publisher: PRI Hasselt
Front cover: Roland Snooks, 'Urban Agency'

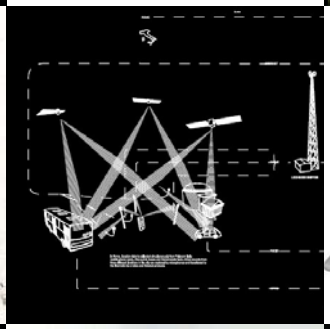
SCRIPT/*E*

Design Game

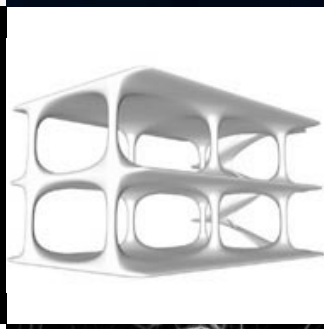
7
FOREWORD



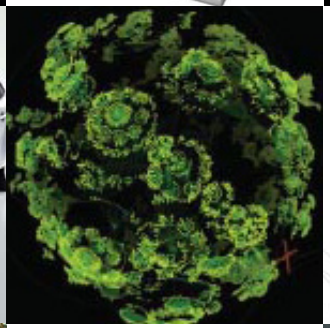


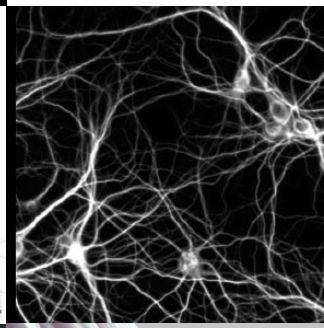


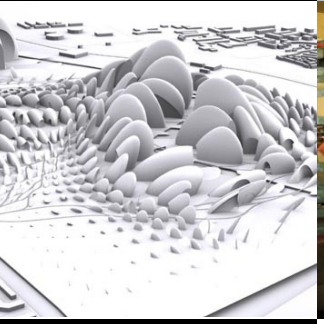
11
INTRODUCTION



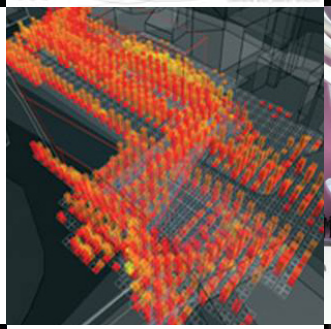
4

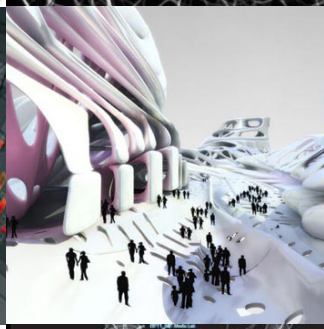




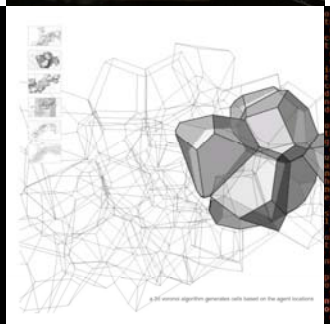


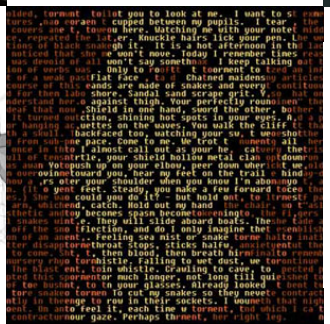
21
PHILOSOPHY

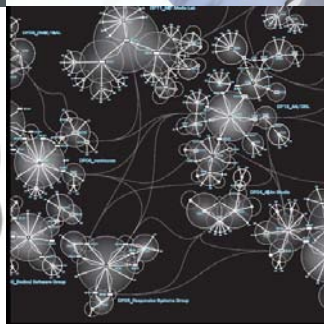


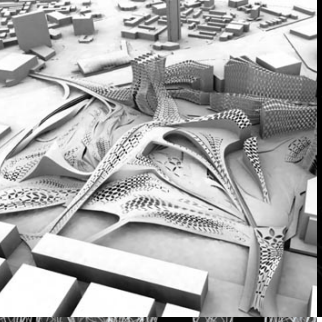
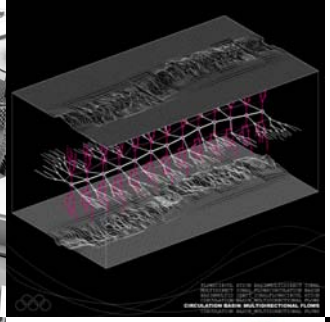


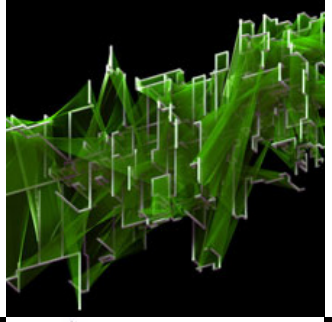
103
INDEX

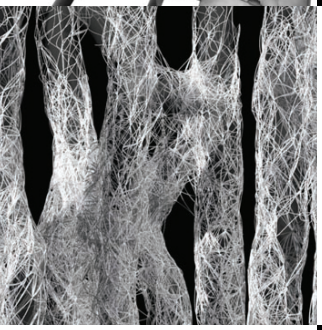


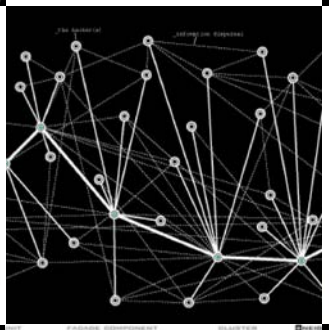


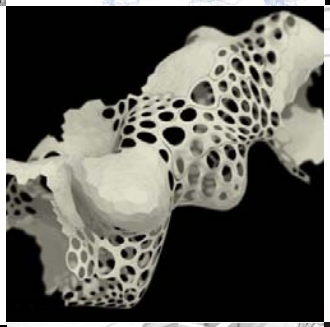


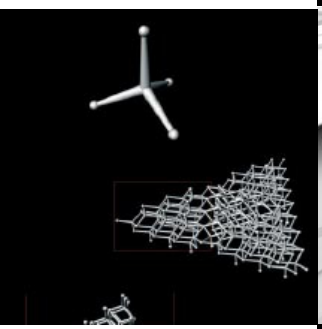
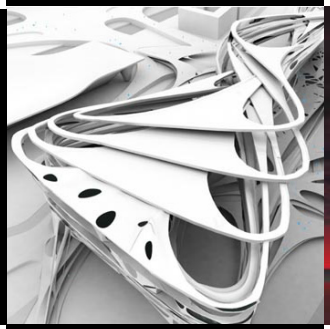





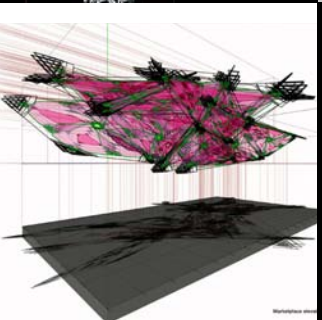


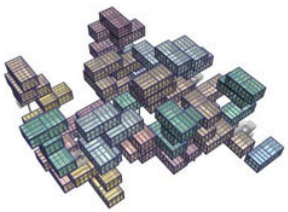
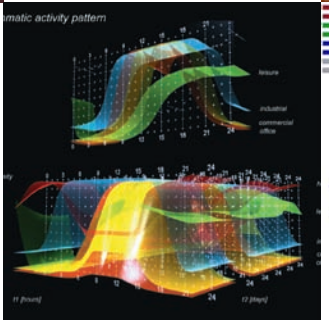


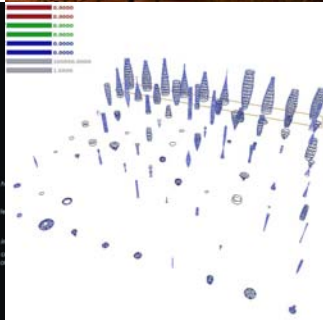










Starter Town
Start New City
Load City
Preferences
Real City Terrain
Exit
MAXIS

7

FOREWORD

SHORT INTRODUCTION

The impulse to the subject for this thesis was an experimental work shop that had been set up within the urban seminar by Oswald Devisch. Within this workshop we worked from the fact that the virtual world and the real world started living more and more beside each other. And so the question was: how far does this resemblance go? By the enormous progress of visualisation techniques in the gaming world and the big successes of social virtual worlds like `Facebook' and `your second life', we started a research on studying the architectural spaces of these virtual rooms. Cause of this resemblance are the evolutionary developments, that we have known the last decades, in the technological industry. The introduction gives an overview of revolutions which have conducted to this electronic dimension, which we, in architectural terms, describe as `hybrid space'.

The virtual world and the real world live more and more side by side. But how far is this similarity in the designs of architects and urbanists? The reason for this development is preceded by recent history. As a result of a number of technological revolutions, an electronic dimension is created which is described in architectural terms as 'hybrid space'. This space is crossed by electronic infrastructures and networks, where alternative cultural and social fields emerges, which often are not visible from a classical point of view. Thus, claims Antoine Picon, the city for example is developing more and more characteristics of the virtual world of a game. There is no coherence. In this way claims Picon, citizens start to behave more and more as `gamers'. And architecture becomes a game played by its users.

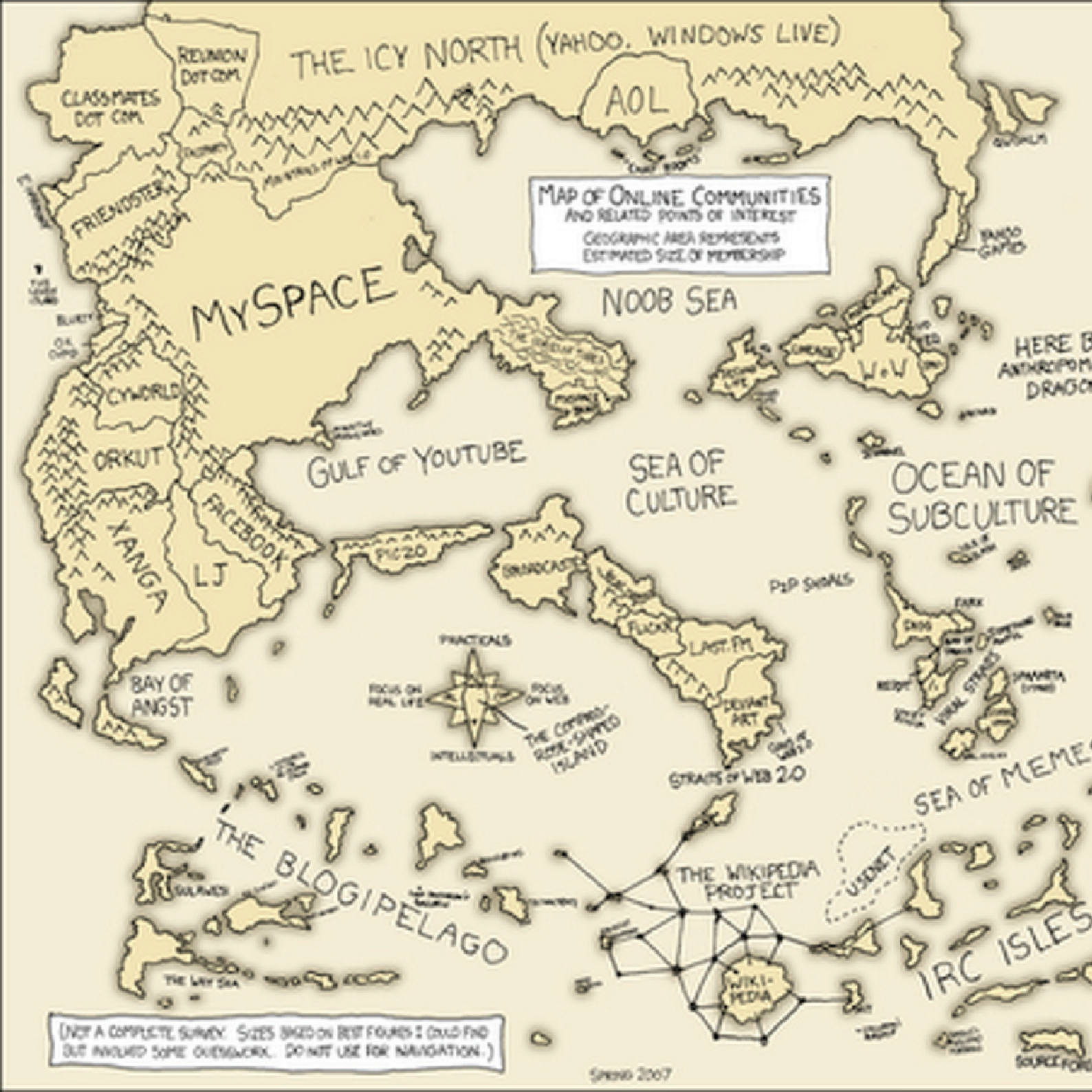
On conferences like 'Game, Set & Match I en II,' organised by K. Oosterhuis, this evolution is examined from the designers point of view. During the design process the game, or architecture , is designed by the architect. This vision forms the underlying thoughts during my thesis research. The title of this script 'design game' refers to the computerscripts that urbanists, architects, engineers and computer programmers develop to create a 'synthetic architecture' which is highly designed within this 'hybrid space'.

Computers have been developed and inspired by the fascination of evolutionary principles which takes place in nature. These principles are described according to the logic of

algorithms. An algorithm is a process of addressing a problem in a finite number of steps. An architectural design process can often be defined in the same way. With this history, this vision and current technologies some progressive planning- and architectural firms are operative in design practice. ONL (Rotterdam, the Netherlands) apply parametric designing on a manner described as file-to-factory. Space Syntax (London, United Kingdom) use computer-based modelling technique to treat cities and buildings.

Throughout all chapters it has become clear that design is a multifaceted process. As such, design tools have often been developed to address isolated issues in design. Through five projects several approaches to different issues are brought to light. To highlight this, a visualization of approaches has been made on which each project can be positioned. Through the study of projects we see that the game, of architecture, is played in the virtual world of the design process. But If the technology of construction systems and materials continuously increases, we can assume that the game, in projects of Kokkugia (Urban Agency) or R&Sie (I've heard about), not only takes place in the virtual world of the design process but even continues into the real world. The question is: will there be a time in which this game is really played or is this vision restricted to what is called an Utopia?

Sueters Bart


THE ICY NORTH (YAHOO. WINDOWS LIVE)
REUNION DOT COM
CLASSMATES DOT COM
AOL
MAP OF ONLINE COMMUNITIES
AND RELATED POINTS OF INTEREST
GEOGRAPHIC AREA REPRESENTS
ESTIMATED SIZE OF MEMBERSHIP
FRIENDSTER
MYSPACE
NOOB SEA
CYWORLD
ORKUT
GULF OF YOUTUBE
SEA OF CULTURE
OCEAN OF SUBCULTURE
XANGA
LJ
FACEBOOK
PIC 2.0
DELICIOUS
FLICKR
LAST.FM
DEVIANT ART
P2P SHOALS
BAY OF ANGST
PRACTICALS
FOCUS ON REAL LIFE
FOCUS ON WEB
INTELLECTUALS
THE CURRENTLY RUSTIC-SHIPPED ISLAND
STRAITS OF WEB 2.0
SEA OF MEME
THE BLOGIPELAGO
THE WIKIPEDIA PROJECT
USENET
WIKI-PEDIA
IRC ISLES
(NOT A COMPLETE SURVEY. SIZES BASED ON BEST FIGURES I COULD FIND OUT INVOLVED SOME GUESSWORK. DO NOT USE FOR NAVIGATION.)
Sphere 2007

1 INTRODUCTION

RECENT EVOLUTIONS

19th Century

By the end of the nineteenth century, the impact of the industrial revolution was widely felt in architecture and urbanism. Newly harnessed forms of power and energy distribution were quickly dispersing and dissolving the pre-nineteenth-century city while new manufacturing and assembly processes were transforming the structural logic, appearance and materiality of the most common building forms. The use of steel and reinforced concrete in multistory, free-span structures gave rise to startling new possibilities in construction. In employing the structural steel frame and Otis's mechanical elevator, Chicago architect Luis Sullivan (1856-1924) pioneered the concept particular to urban life in the twentieth century: **the vertical city**. Nascent communications and media technologies, such as telegraphy aerial photography and cinema, reshaped humanity's sense of geographic distance, proximity and temporality. More significantly, the advent of **the car** – a kind of horizontal elevator – brought about an even more radical transformation: an extreme form of urbanity that would spread itself over national networks of highways, secondary roadways and transportation interchanges. In the first half of the century, such modern visionaries as le Corbusier, Frank Lloyd Wright and Mies van der Rohe would seek to devise appropriate architectural forms and urban design strategies to address the technological advances and modes of urban organization.

At the close of our century, it is the information revolution that is metamorphosing architecture and urban design. Digital technologies are transforming the nature and intent of architectural thinking and creativity, blurring the relationships between matter and data, between the real and the virtual and between the organic and the inorganic and leading us into an unstable territory from wich rich, innovative forms are emerging. A new time-space vernacular is rescripting the model of the city as cable and satellite connections span massive physical distances along a curved terrestrial geography. Objects, places, people, buildings and cities are no longer framed in the moment but instead approached along multiple and associative routes. Through visual and non-visual means of mobile cognition – satellite-imaging, electron-scanning or heat-sensing – structures and buildings are being set free from a conventional linear viewpoint. Buildings can become less icons of fixity and immobility and more like inclusive fields of organized materialization.

"Buildings can become less icons of fixity and immobility and more like inclusive fields of organized materialization."

Technological revolutions

20th Cen.

The cultural and social revolution brought on by telecommunications and information technologies is rapidly transforming the field of architecture. We live in an era of accelerated change, in which data speeds invisibly around the globe and the flow of information has superseded material exchange, and complex digital infrastructures have inscribed themselves within our recognizable mechanical and urban patterns. Metropolitan populations, previously culturally bound and physically localized, have become nomadic and transitory, following the dynamic movements, accumulations and diffusions of international capital investment and diversification.

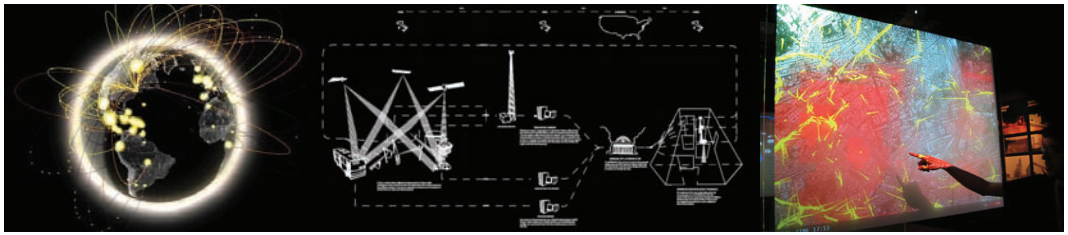
Our international telecommunication networks have become characterized by agitated, irreversible super-connections that operate outside conventional human understanding of time and space. We no longer communicate with friends, family or associates exclusively in a particular place; rather, we communicate both in the local context and across time zones and cultures. A virtual world of international connections has replaced the local spots by a space crossed with electronic infrastructures and networks from which alternative cultural and social domains.

21st Cen.

Against this shifting background, architecture itself is mutating, redefining its boundaries, its essential codings, to adjust to an increasingly supple and volatile world. Architecture is recasting itself, becoming in part an experimental investigation of topological geometries, partly a computational orchestration of robotic material production and partly a generative, kinematic sculpting of space. These new modalities for architecture, the topological-

geometric, the industrial-choreographic and the generative-kinematic, are united transversally by the notion of an hybridized spatiality. Hybrid space defines an architecture that is produced by breeding ideas of concepts of contrast and heterogeneity – the strong and the weak, the formed and the formless, the real and the virtual – and that evolves through the embodiment of competing identities – unraveling and consuming opposed geometries and spatial postulates. This new architecture organizes the world by arranging the spaces between things rather than perpetuating the myth of ideal form. Hybrid space unbinds an architecture of inclusion and absorption, recombination and admixture.

The 'hybrid space' mixes the **globalizing** liquid "soft architectures" of digital media flow over, under and through the local, concrete and "hard architectures" of our contemporary cities creating an indeterminate, "floating" environment, an interface between public and private, collective and subjective, provincial and planetary. Some architects in this thesis claim this ambient, symbolically rich and multidimensional world-space as an extraordinary context for architectural exploration. Some projects endeavour to instrumentalize the simultaneous identities of the global and the local experience. As Manuel Castells has pointed out, "networks do more than organizing activities and sharing information. They are actual producers, and distributors, of cultural codes." In the place of merely representing these projected identities and territories (the global or the local), hybrid space proposes an approach to architecture that creates new **cultural codes** and modalities – turbulences or disputations within the physical and electronic networks that connect our international and local cultures.



Sense*able* city lab, MIT

"Architecture is recasting itself, becoming in part an experimental investigation of topological geometries, partly a computational orchestration of robotic material production, and partly a generative, kinematic sculpting of space.

City

The four urban functions of working, living, leisure and transport which Le Corbusier once so elegantly deployed in his model of the city can no longer be separated from each other either spatially or socially. Living and transport have become practically identical. What was once a place, the **city**, has now become a brand, a logo, a "townscape" which itself consists of clusters of brands and logos. The city has ceased to be a clearly localizable spatial unit and has transformed into what might be termed an "urban field " a collection of activities instead of a material structure. The Contemporary urban experience is splendidly symbolized by the cell phone: wherever your mobile works is city, and anywhere else is countryside.

A city is an **unstable system**, a living system which is in a state of continual decomposition, but which also continually reorganizes and rearranges itself, which expands and shrinks. One of the actors or "agents" in this process of self-organization is the urban population, including the city's architects, urbanists and local government officials. Other "agents" include technological developments, the mass media and migrations. What is wrong with the various "post-" terms is that they describe the city from the outside, from the perspective of the past. But every description of a process is itself a product of that process. Every cityscape is a function of the city imagined. If you want to understand a development, it's no good standing outside the process; you have to wade into it. You have to allow yourself to be developed by the developments. From the outside, you see only the movements: what stands still, what shifts, what disappears. From the inside, you detect the transformations: what direction things are going in, what is changing and what new things are emerging.

Trans

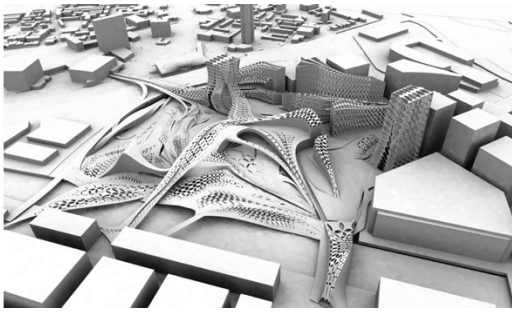
Cities are growing increasingly complex, increasingly rich in internal and external linkages, increasingly comprehensive and concentrated, increasingly transparent yet incomprehensible. That's obvious as soon as you abandon the "post-" position and move to a "**trans**-" attitude – in other words, when you consciously go along with the developments instead of frantically trying to maintain a position outside them. People don't change because they wish to do so, but because they allow themselves to be changed and, in doing so, themselves modify the broader process of transformation in which they are being swept along. "Transurbanism" is by contrast urbanism plus transformation. Transformation is the multiplication of information. Transurbanism is a theory of transition of cities as they are now, towards a design process in which the highly informed character of every built environment is used

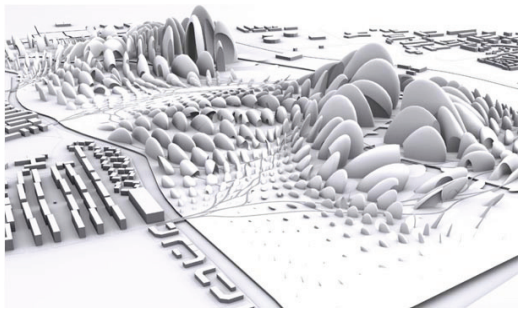
"What was once a place, the city, has now become a brand, a logo, a "townscape" which itself consists of clusters of brands and logos.

as a design resource by that environment itself. The high-informed nature of a building is conducted by the 'hybrid space' in which it is located.

Transurbanism is urbanism in the era of globalization. The design challenge for architecture in this context is, instead of trying to create a single public domain, to create an atmosphere for the establishment and coexistence of a diversity of public domains. Transcontextualize. You cannot design a city, but you can help a city organize itself as a living structure – not by breaking down all barriers to the streams of information and commodities, but by allowing specific obstacles, channels, retardations and accelerations to be designed for individual streams, and thus to be **informed** by the city itself. The apparatus that makes it possible to pursue this kind of design practice has matured in the past 40 years. The technical basis underlying the narrative of globalization is, after all, the computer, the "great communicator," the "great interactor."

If the seminal avant-garde of the early twentieth century designed an architecture for the Machine Age, then the architects featured in this thesis are devising transformative, poetic and pragmatic responses to the technologies, urban networks and post-mechanical processes of the Information Age. They are developing **spatial routines** and **urban codings** for a world that is at once unfixed and fixed, here-there and there-here, dislocated and located. Theirs is neither a revolutionary nor utopian architecture but an architecture of evolution, contextualization and transmutation. Their researches are triggering a phase-shift in our perception and comprehension of space, materiality and time at the start of this new age.





AAdrl Loden, UK

"Transurbanism is a theory of transition of cities as they are now, towards a design process in which the highly informed character of every built environment is used as a design resource by that environment itself."

Perception in virtual worlds

About a different view and **perception** of space Antoine Picon claims that the city for example is developing more and more characteristics of the virtual world of a game. There is no coherence, as in a computer game you roll through a secret door of a Roman amphitheatre into an underground lava cave, in the same way you go past a shopping street and around the corner you suddenly stand between some office buildings. In this way claims Picon, citizens start to behave more and more as `gamers'. And architecture becomes a game played by its users. On conferences like 'Game, Set & Match I en II,' organised by K. Oosterhuis, this evolution is examined from the designers point of view. During the design process the game, or architecture , is designed by the architect. During the life cycle of a building, and the built environment, the game is played by its users. And not only by its users but also by the building itself. Hereby several **configurations** are developed on which the building reacts and reconfigurates itself in content or form or I both ways. Just as in the virtual world of a game where the gamer controls his actions on witch the game continually reacts. If the technology of construction systems and materials continuously increases, we can assume that such processes not only take place during the design process in the virtual space but would even take place in the real space of the built environment. This vision forms the underlying thoughts during my thesis research and has been described in the chapter philosophy.

Developing a game of architecture in such way is not simple and requires another approach of the design process. If the architect designs a game, is this still designing? This brings us back to the question: what is **designing** and what does it stand for? As designers we have the capability to plan, to look into the future and therefore anticipating on what will happen. On the other hand we base our designs on what once was, for example archetypes. This contradiction is treated in the chapter theory. If we study archetypes more closely as refer-

ences in our designs, it leads to a point in geometrical terms called **topology**. Topology is a branch of mathematics that deals with the properties and relations between objects which remain intact after deforming these objects. These characteristics are explicitly used in CAD. Deforming or manipulating geometrical objects is possible in programmes such as 3d Studio Max, through an interface where several variables can be changed. This technique is better known as **parametric designing** and the last decade it hasn't missed its purpose in the construction industry. These days CAD's offers applications to designers which give them the possibility to write and execute scripts. Because computers have been initially developed, and they still do, from a fascination for **evolutionary principles**, like for example growth processes, biological science have played a huge role in its development. Current generation of designers and architects has not forgotten this and ever since it has lead them to incorporating, like we call it these days, evolutionary computation in the world of architecture.

Growth processes or other biological processes, for example behaviour of ant colonies, can be described abstractly according to the principle of genetic algorithms. An algorithm is a process of addressing a problem in a finite number of steps. It is an articulation of either a strategic plan for solving a known problem or a stochastic search towards possible solutions to a partially known problem. In doing so, it serves as a codification of the problem through a series of finite, consistent, and rational steps. In the chapter 'techniques' several examples are presented which are generated by algorithms,. Through an architectural design we analyse the codes of algorithms written according to a **script** language. As such, numerous design systems have been described according to the linguistics of scripts which work entirely autonomously, so without human intervention. The goal of these design systems is, also described as `**micro worlds**', to develop new external systems of representation that foster more effective learning and problem solving.

With this history, this vision and current technologies some progressive planning- and architectural firms are operative in design practice. ONL (Rotterdam, the Netherlands) apply parametric designing on a manner described as file-to-factory. Space Syntax (London, United Kingdom) use computer-based modelling technique to treat cities and buildings. Throughout all chapters it has become clear that design is a multi-faceted process. As such, design tools have often been developed to address isolated issues in design. Through five projects

"We must remain watchful of the machine's ruinous endgame played out as urban forms, space and relations."

several approaches to different issues are brought to light. To highlight this, a visualization of approaches has been made on which each project can be positioned. Through the study of projects we see that the game, of architecture, is played in the virtual world of the design process. But If the technology of construction systems and materials continuously increases, we can assume that the game, in projects of Kokkugia (Urban Agency) or R&Sie (I've heard about), not only takes place in the virtual world of the design process but even continues into the real world. The question is: will there be a time in which this game is really played or is this vision restricted to what is called an Utopia?

21 PHILOSOPHY

VISION

Synthesis

The technical and social developments thus affect the nature of buildings and the perception of the architecture. As a consequence of recent advances in natural science many ongoing developments in architecture have been redefined. The design territory of urban development is expanded with a virtual world, the '**hybrid space**', parallel to the real world. This space is created by the communication and information technology which recent evolution is due to the electronics industry. The incorporation of electronics in our lives made it is possible to enter this 'hybrid space'. The electronic revolution inspires a '**synthetic architecture'**. The notion of synthetic can be explained in two ways: as a smelting of different arguments into a new whole, and in the sense of artificial, assembled in the laboratory. The synthetic dimension is a locus of activity that has been staked out anew using the latest available techniques. The notion of synthetic can be explained in two ways: as a smelting of different arguments into a new whole, and in the sense of artificial, assembled in the laboratory. The synthetic dimension is a locus of activity that has been staked out anew using the latest available techniques. In the history of architecture and the visual arts there have regularly been situations where art and science have experimented, each in their own way, with a new field of activity.

We are now in the throes of a revolution, one that will shake architecture free of its foundations. The electronic revolution is penetrating deep into our daily lives. Fossilized social ideologies are left helpless by the implanting of electronics in the lives of billions of people. The ideologies of socialism and capitalism were derived from the industrial revolution. The electronic revolution will give rise to a new ideology, one that can no longer call upon material hardware. Electronics and programmatic software are accessing new synthetic dimensions, new conceptual areas, new fields of activity for visual representation. The ideas that gain shape in these new realms of thought form the immaterial substance of future new ideologies. They create the basis for a new architecture, redefined from **bottom-up**.

Electronic

The electronic revolution has triggered a new aesthetic. Mechanical functioning is now taken for granted rather than emphasized as such, as was the case. Functioning is no longer a visible act. The aesthetic of the telephone began as purely ergonomic, becoming purely stylistic

"During its construction that building is briefly at the hub of global economy.

at a later stage. Today's cars are no longer made up of distinct, individually identifiable components. All their functions have gelled into a **cohesive volume**. The chassis has been replaced by the monocoque construction, lamps, mirrors and bumpers have been absorbed in the overall form. Now the whole idea of the car is at the threshold of electrification. Doors are opened by remote control with infrared signals, the dashboard gives spoken information, road maps have been digitized, with sensors scanning and responding to the roads.

By analogy, buildings are developing from statically determinate to statically indeterminate structures, where all the component parts are tuned to one another. As a product, a building is a complex unit, a long way past the simple stacking of standard elements. In programmatic terms buildings have been freed from their primary functioning to put up ever greater performances in an increasingly invisible way. The electrified building will be able to respond ever more effectively to particular wishes regarding physical comfort, and to the immaterial wishes of individual users.

At the same time the smart building will take a more economical position on energy. We can define an architectural object as a cell with a semi-permeable wall that is part of the **ecosystem**. A building is autonomous only up to a degree. Static in itself, it is informed by a changing context both inside and out. The weather conditions are forever changing. The semi-permeable cell membrane permits only the most selective exchange between interior and exterior. The building absorbs energy from outside and in turn emits energy. Energy in the form of information, radiation, electrical voltage, heat. Those who use the building are messengers, importing and exporting information in an unending cycle. During its construction that building is briefly at the hub of global economy. It forms, temporarily and locally, a small point of stimulation in the global network of artificial, humanly conceived activities. An architectural product is a point of condensation in an ecological process. To indicate that this process is completely fabricated, we can speak of an artificial ecology. The architectural building is becoming more and more aware of electronics in which all of the components are developed in close conjunction with each other. The enormous amount of variables and parameters of this 'synthetic architecture' is controlled by **computer systems**.

Architecture has become an industrial product, partly because of the computer. A major consequence of CAD/CAM applications is that the component parts of the building can be custom-made without bother. The manufacturer no longer produces a generally applicable product, but supplies a particular product with a particular make-up. No longer a question of compiling components from a catalogue, architecture can now become an industrial product in its own right, a synthetic whole of attuned parts. As a result of its malleability, the open volume ushers in a new architectural idiom. Le Corbusier's open plan, 'le plan libre', assumes a concrete skeleton within which partition walls can be freely placed and freely formed. The open volume theory, for its part, proceeds from the possibility of freely shaping the volume as a whole and, as an object, placing it freely in space. Architecture is then the designing of a membrane between a freely subdivisible external space and a freely subdivisible interior. This ability of the volume to be freely subdivided does not assume a flat floor upon which walls can be placed. The volume is spatially broken open. The interior becomes a three-dimensional construction inside a given volume . The architecture of the membrane can be fully unhitched from that of the interior. A like reasoning obtains for the context in which the open volume is to be placed. The spatial concept of today's urban design is closely tied up with the architecture of the buildings themselves. Here too the open volume would enable these to be fully disconnected. Then the urbanistic rules of play would have no further influence on the visual outcome of architectural objects. The open volume would then manifest itself as a cell in the three-dimensional space of the urbanistic organism.

DRD LAb, Stuttgart

"Architecture need no longer be generated through the static conventions of plan, section and elevation. Instead, buildings can now be fully formed in three-dimensional modeling, profiling, prototyping and manufacturing softwares, interfaces and hardwares, thus collapsing the stages between conceptualization and fabrication, production and construction."

It is instructive to confront Le Corbusier's **five points** with the, in part hypothetical, hallmarks of today's synthetic architecture. All five, of great relevance in the early part of the last century, need revising in their entirety.

1 The structural skeleton, bearer of material functionalism, has been synthetized as a self-bearing, electrified and stylized building body.

2 The ideal of the roof garden has been reduced to a romantic detail in view of the explosive evolutionary success of the artificial furtherance of the global ecosystem.

3 The open plan has been pulled up into an open volume. Whereas plan and elevation can do no more than illustrate a flattened and discontinuous space, the new synthetic architecture is given shape directly in virtual, fully three-dimensional space.

4 The horizontal window arrangement has been trivialized by the introduction of the smart membrane, a selectively permeable wall.

5 The free facade has been further radicalized into the thoroughly autonomous design of the open volume.

DRD LAb, Stuttgart

"The open volume would then manifest itself as a cell in the three-dimensional space of the urbanistic organism. Building and with it architecture are freeing themselves from their static harness. Buildings too are becoming part of distributed networks, during the developmental stage, in the production process, and above all during their functioning as information processing bodies.

Internet

The entry of electronics within architecture and the development of the World Wide Web have ensured that the built environment continuous exchanges **information**. Architecture has always been a medium to exchange information. People do not have to move anymore, the electronic dimension of architecture does this for her. The new information resides distributed in one or more of the millions of interlinked computers, and in turn forms fresh input for other users in the network. The Internet functions as a gigantic worldwide neural system. What influence does this have on building and architecture?. For instance, we are already witnessing a change in the production of building materials from linear, centrally run extrusion processes to programmable, data-run transformation processes. Building and with it architecture are freeing themselves from their static harness. Buildings too are becoming part of distributed networks, during the developmental stage, in the production process, and above all during their functioning as information processing bodies. The design of the bodies of buildings is becoming a **data-driven process** and the life-cycle of those bodies is directly linked to the databases. Real-time communication gets under way between the various data-driven and data-processing bodies. Architecture becomes a push-and-pull medium.

NOX, D-tower, Doetinchem, the Netherlands

"D-tower is a coherent hybrid of different media, where architecture is part of a larger interactive system of relationships.

Real-time

The synthetic architecture of the project D-tower continuously calculates fresh data in relation to neighbouring data. Swarms of real time data are the parameters inside the computer scripts. The formula's plus the parameters are building the emotional factor of that body in real time.

'The building body finally goes wild.'

Kas Oosterhuis, ONL

Data

The 'synthetic building' that is out of control cannot be seen in isolation from an environment that is out of control too. The out-of-control building is an information processing machine that develops in an information producing environment. To understand that better, we first have to look at the ecology of a built environment that is changing in real-time.

In New Babylon (1958-1972) by the artist and architect **Constant Nieuwenhuys**, our bodies move within a constantly changing environment. In New Babylon virtuality has penetrated the material, the sectors are data-driven and programmable, the structures and atmospheres react to the nomadic users. If the gypsy user were ever to return to the same spot, the environment would have changed completely in form and content. The yellow sector would still be yellow, but in a different physical form and with a new information content. New Babylon is actually the temporally condensed version of our society. In his essay 'Next Babylon, Soft Babylon', Marcos Novak notes that New Babylon has now been realized in the infrastructure of the global Internet, the wireless telephone network, and the constellations of satellites that bring the whole earth within wireless electronic reach. While Greg Lynn states that the character of the abstract nature of the new technologies has to be understood and intuitively operated before it can be translated into architectural form (Greg Lynn, Animate Form), this new architecture is in fact already developing rapidly on the Internet. The Internet is itself developing because all of the participants in the network develop. This real-time process of evolution indicates the direction that the development of physical buildings will presently take. The essence of real-time technology is that it wants to stay real-time. It is the will to live. The building becomes a construction under development in an environment under development, played by users who are in movement. Stasis is therefore an extremely slow form of motion. Architecture must not be confined to the extremely slow movement, but should choose its own tempo on a sliding scale from extremely slow to the speed of light.

"First we need that body, not your human body, but that product body. Than that body needs a skin. The body becomes an instrumental playstation.

The architecture of the future will calculate its construction in real-time, determine its position in relation to other constructions, continually adjust its relation to its users on the basis of a constant flow of new data from the database that upgrades itself in real-time. The building sets out on a course of its own, and is steered by its users. The building knows its users, it is aware of individual differences and preferences of its users. The user plays the building like an **game**. Now the game becomes programmable and its operation so complex that behaviour becomes unpredictable (cf. Lynn's pet). The behaviour of buildings becomes surprising and emotional. Once a database has been created, the figures can be processed in a great variety of ways in computer scripts which can drive not only the geometry of the design process but also the behaviour of the realized design in time (light, sound, temperature, change of shape, functioning).



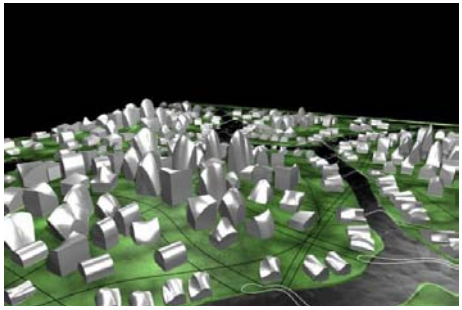
New Babylon, Constant Nieuwenhuys

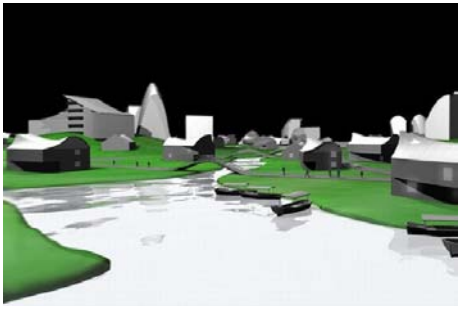
Game, Set & Match

Architecture becomes a **game** being played by its users. And not only architecture will be subject to the forces of real time calculation. Also planning, construction, interior design and landscape design are ready to be developed as real time games. During the design process the game is designed by the architect and played by all parties involved. During the life-cycle of the building and the built environment the game is played by their users, by the visitors and by the built environment itself. Visitors become participants in the experience economy. By playing the game the participants **set** the parameters. Each actor triggers an array of sensors writing the new data to a database, from where the building picks up the new data and starts reconfiguring itself, in shape, in content, or both in shape and content. Then the new configuration will **match** to the desired conditions. It is fair to say that the building will find itself in a state of continuous operation. The building, consisting of numerous cooperating programmable elements, will behave exactly like a swarm. The building elements will show flocking behaviour, always keeping an eye on the neighbouring actor, always ready to act

"Now the instrument becomes programmable and its operation so complex that behavior becomes unpredictable."

and react. Hence Kas Oosterhuis proposes the new motto for the profession of architecture: “Game, set and match”. To be played over and over again.





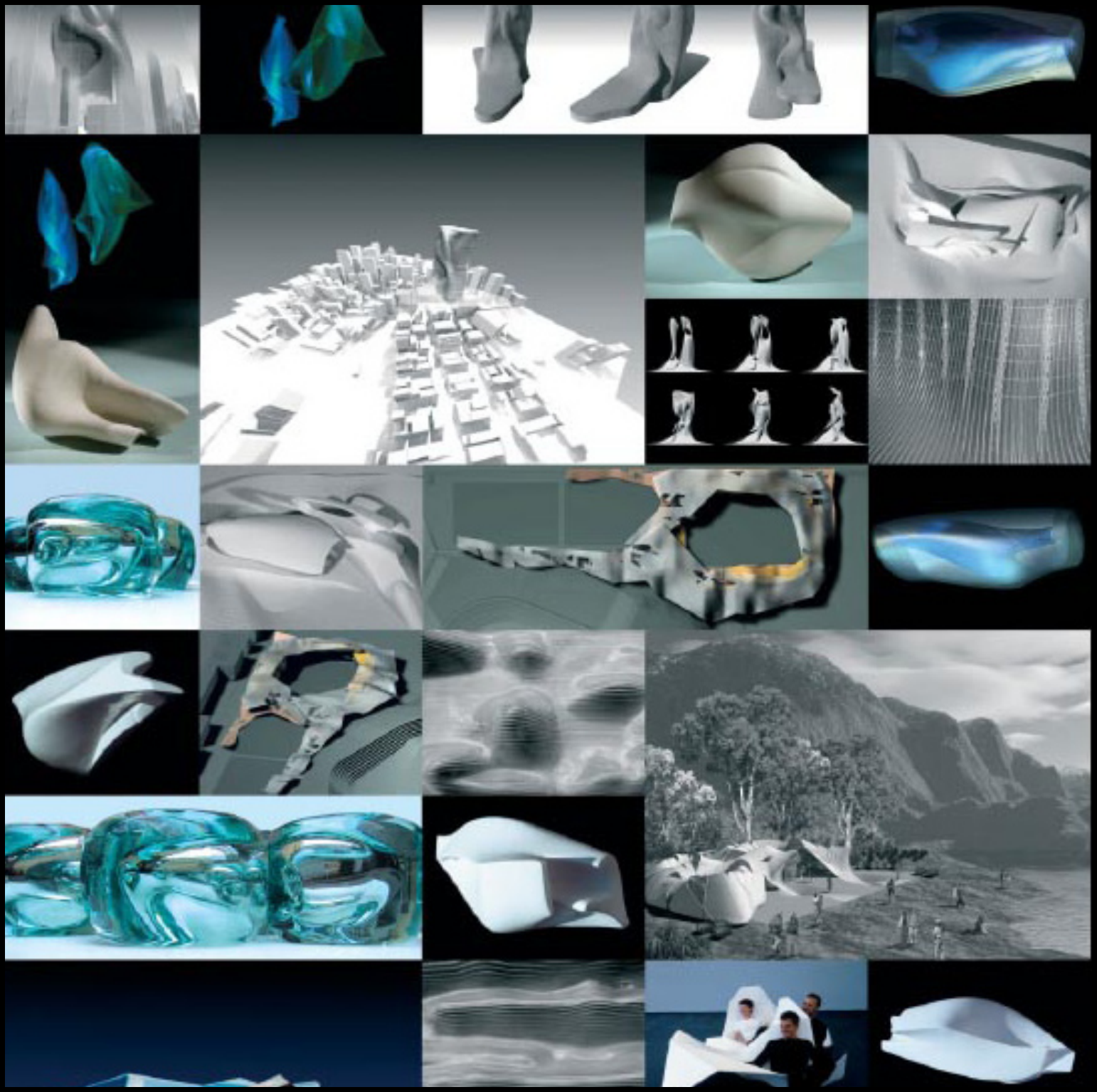
Masterplan, New Can Town, Zhujiajiao, ONL

If Le Corbusier, during the era of the Industrial Revolution, gave shape and meaning to architecture in the real world then form and meaning is now given shape within the ‘hybrid space’ of the Digital Revolution. Virtual reality, including all software ever written for any platform, is hyper-real. Simply because we know the stuff where it is made of. We know every bit and byte. In the Digital Revolution reality has been re-written from ground zero. All matter, including all material where architecture is made of, is being redefined as information flow. Matter is information, architecture is information. Architecture always was information, but now we start to work with architecture as information.

“If you are not in real time, you’re dead”

Kevin Kelly

If that is the case then traditional buildings are not here with us in real time, they merely function as a dead bodies in the background of human activities. But now we have found the tools to activate the building. Not only can we activate the lighting conditions and the sound environments, or generally speaking the content, but we can activate the very structure of the building.



31 THEORY

DESIGNING

Designing

Architecture developing as a '**game**' in real-time is not easy and requires a specific approach of the design process. If the architect designs a game, is this still designing? This brings us back to the question of what exactly is designing, and where it stands for? Design is about conceptualization, imagination, and interpretation. Design is vague, ambiguous, and indefinite process of genesis, emergence, or formation of something to be executed but whose starting point, origin, or process is often uncertain. Design is about the spark of an **idea** and the formation of a mental image. It is about the primordial stage of capturing, conceiving, and outlining the main features of a plan and, as such, it always precedes the planning stage.

Meaning

According to the Greeks meanings of the word design, design is indirectly linked to a loss of possession and a search into an oblivious state of memory. This linguistic connection reveals an antithetical attitude towards design that, in Western culture at least, is about stepping into **the future**, a search for new entities, processes, and forms, frequently expressed by the terms novelty or innovation. If, according to the Greeks, design is about something that we had but do not have any more, hence it is lost somewhere in **the past**. What is its connection to something that is about to become the future, i.e. a novelty? Why would they bring up such an unexpected and obscure relationship? Is it possible that the novelty in the sense that we understand it today, according to the Greeks, does not exist per se and anything new is just an illusion?

If we look deeper into pre-Socratic philosophers such as Xenophanes, Parmenides, or Zeno, one common agreements between them was the assumption that nothing comes out of nothing and nothing disappears into nothing; nothing can just pop up or vanishes without a trace. Change is nothing but a transformation from one state to another. According to this logic, design as a mental process of creation can be seen as bounded by the limits of preservation: any newly conceived thought, process, or form is nothing but a reordering of previous ones.

However, if we consider this possibility, then we are confronted with the problem of origin: as every "new" idea depends on its previous one, then there must be an origin, a starting

"The primitive, eternal, and universal nature of archetypes serves not only as a point of departure but also as a point of reference.

point, a root or roots out of which everything spurs, tangles, and multiplies offering glimpses of what appears occasionally to be "new." Hence, we are led to the conclusion that the origin must be fixed, eternal, and indestructible. And since novelty involves the negation of existence (i.e. something that did not exist before), novelty is impossible. It is only a sensory illusion.

Archetype

The notion of **an origin** is important when discussing the process of design. Because of its investigative nature design is always associated with a starting point, a pivot, out of which style, fashion, or mannerisms results. That starting point is important for at least two reasons: first, and most obvious, it serves as a pivotal point of reference that identifies, categorizes, and determines a wide range of similar products. Second, and less obvious, is the fact that an origin belongs to the distant past and as such it involves the reminiscence of something that was lost but whose consequences are still present. While memory is usually about mundane, common, and ordinary past events, it is also about that which is lost in the distant past, the primordial, archaic, and primitive. The origin as such, is elusive, evasive, and indefinite yet it is always present in the form of a sign that points out at the increasingly distant past. Thus, in searching for the origin one is challenged to seek the basic, archaic, and primitive qualities of the first encounter: **the archetype**.

- Evolution! - Homology - DNA structure - Chromosome

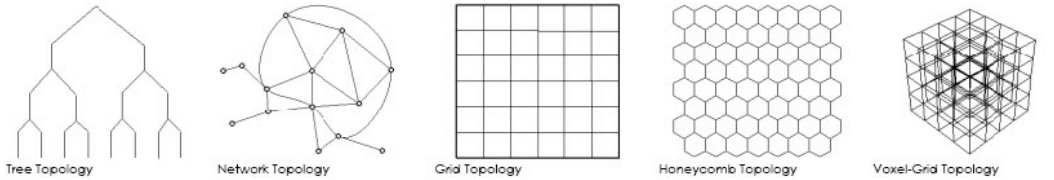
Design Tooling; S. Dritsas, A. Kilian & Y. Loukissas

Geometry

First, as designers we have the capacity to plan, to look into the future and thus to anticipate on what will happen. Secondly, we base our designs on what once was, for example archetypes. If we apply archetypes as a reference to use as an underlaying sheet, it leads in terms of geometry to certain topologies. The topology occupies itself with the properties

"The act of forgetting is not a submersion into oblivion but rather the erase of false connections and the return to the origin.

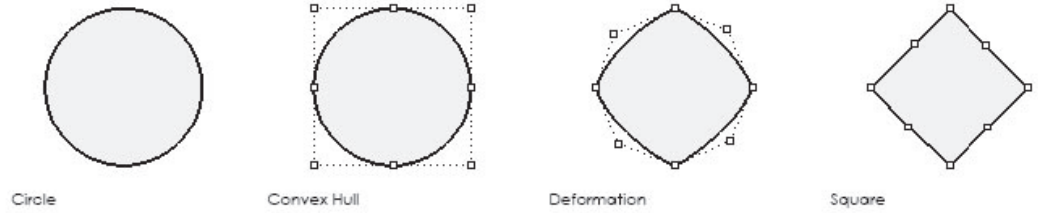
of objects which remain kept after deformation. Topology is a method of describing information based on relative rather than absolute positions. A topology is specified in terms of connectivity and adjacency between objects. The topologies of a tree, lattice, network are usually described by nodes and a connecting links whereas grids are described by adjacent/neighbouring cells.



Design Tooling; S. Dritsas, A. Kilian & Y. Loukissas

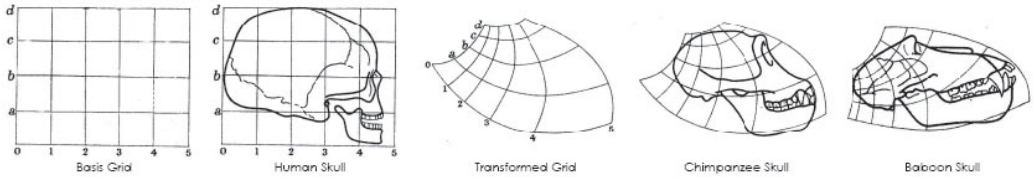
Topology

One definition of topology is the topographic study of a region: the survey and graphical delineation of a place and its configurations, elevations and positions. A topology shows the **relationship** between things, man-made and natural, on the earth's continuous surface. Topology is also the branch of mathematics that investigates geometric configurations (as a point set) that cannot be altered if subjected to one-to-one transformations by shrinking or enlargement. For instance, if one tied a piece of string around a sphere and shrunk the sphere and the loop of string, eventually the loop would diminish to a point. Shrinking the sphere will not prevent the collapse of the loop; a sphere is therefore not a topological object. On the other hand, if one tied a loop of string around a torus and reduced the doughnut and the loop, the loop would never shrink to a point: the torus hole would prevent the loop from collapsing. A torus is thus a topological object. Topology involves the study of strange surfaces that can be transformed without collapsing or breaking because of the rubbery structure of their surfaces. Topology is an abstraction or generalization of geometry; while objects are "rigid" in Euclidian geometry their topological counterparts are "flexible". A circle, a square and a triangle; a straight line and a curve, for example are topologically equivalent. Topology is also known as the geometry of the rubber sheet.



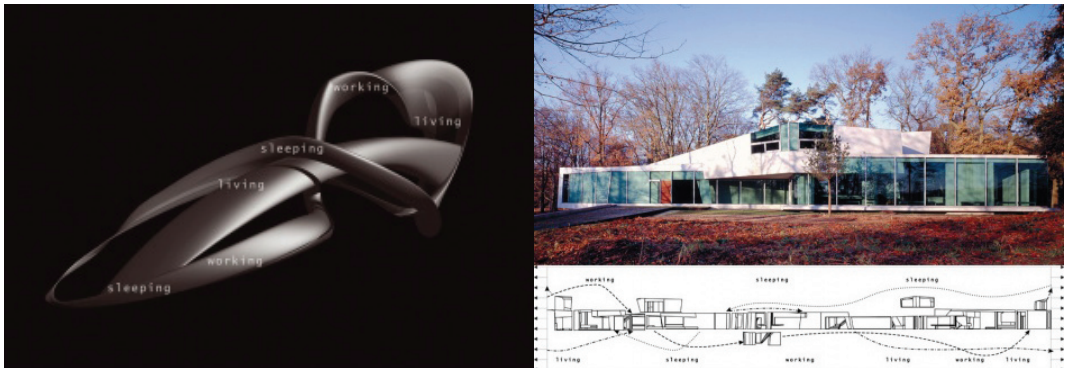
"Topology is an abstraction or generalization of geometry"

A third definition of topology is the anatomy of a particular area of the body, the form and qualities of an organ, for example, or the figure and outline of an organism. D'Arcy Thomson used topological transformation in describing the formal relationships between species. In the following images for instance, a human skull under a topological transformation comes close the the skull of a chimpanzee. The role of topology in this instance is one of an underlying structure that remains behind the phenomena, but establishes a continuity and relation between them.



Transformatie van een schedel, D'Arcy Thomson

All of these definitions of topology are essential to the computational architectures collected in this thesis. These explorations of built form are not based on the pure Euclidean geometries of the sphere, cube or pyramid but instead are often modeled on the torus, the Möbius strip. In the process, this architecture affirms **the paradox** of topology: a continuous looping into and out of back and forth, on a surface without end or beginning, which has neither interior nor exterior, but which is always experienced as a strange entity.



Mobius House, UN_Studio

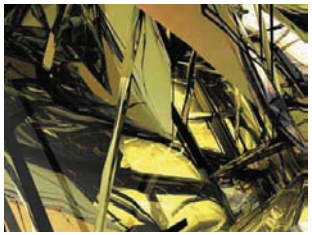
"Architecture is becoming like "firmware," the digital building of software space inscribed in the hardware's of construction."

Parameters

Topology is a branch of mathematics that deals with properties and relationships between objects which remain kept after deformation. These characteristics are specifically used in CAD programs. Deformation and manipulation of geometric objects is frequently applied in programs like 3D Studio Max, where through an interface different variables can be adjusted. This technique is better known as parametric designing which hasn't missed it's influence all over the construction industry for the past years.

Parameters can form the basis for powerful concepts of architectural forms by describing a range of possibilities. Fixed values in a process replaced by variables, single by plural. Because of parameters designers can create an infinite number of similar objects. Often these geometrical expressions of a postulated diagram of variables are related to for example dimensions and relations. With parametric designs it are the parameters of a design which are fixed, not the form. Calculations can be used to describe relations between objects. From that point on we speak of an **associative geometry**.

Parametric designing often contains an algorithmic description of geometry. Thus Marcos Novak constructs in its `Data-driven-forms and `Paracube, by means of Mathematica software, mathematical models and generative procedures which are limited by several variables. Each **variable** is a field in which an external influence can be placed, either static, or dynamically. When we use this manner of work, we conclude that we are less committed with the manipulation of objects themselves, but more with the manipulation of relations, fields and the bending of the space. As so this means parametric design does not necessarily have to produce stable forms.



Data-driven Forms, Marcos Novak

"Architecture need no longer be generated through the static conventions of plan, section and elevation."

A proposition which Greg Lynn examines in `**Animate form**' is that with conceptually parametric designing the presence of contextual information fields is as important as the presence of the internal parametric system. In other words the architectural form is not only an event of its internal, parametric-driven relational logic. He also aims on answering to the dynamic, frequently variable, influences of its surroundings and socio-economic context. From a conceptual view architectural form can hereby be a highly plastic, mutable entity that interacts with external strengths. In this way the context of a design becomes an active abstract space which is conducted by a number of strengths. These strengths are stored in the form of the design.

From a parametric view design will, when it is consequently applied from the conceptual phase to realization, have widely consequences in the construction industry, as well as in the role which is played by the architect. For the first time in the history architects will not design a specific form of a building, but he will set out **a range of principles** in a sequence of parametric calculations through which instances of the design can be generated and modified when necessary. Parametric designing calls a stop to fixed solutions, and hankers to an exploration, to an infinite number of possibilities.

'Synthetic architecture' needs no longer be generated through the static conventions of plan, section and elevation. Instead, buildings can now be fully formed in three-dimensional modeling, profiling, prototyping and manufacturing software's, interfaces and hardwares, thus collapsing the stages between conceptualization and fabrication, production and construction, numerical data formations and spatial experience. The unique character of handwork and systematic mass production can now commingle in CAD/CAM mode of creation, which can produce series-manufactured, mathematically coherent but differentiated objects, as well as elaborate, precise and relatively cheap one-off components.

As the French architect, technologist and theorist Bernard Cache has argued, architecture today should be understood as an "electronic technical art," based less in the representation of the ideal forms than in the scripting of machining codes and routines for numerically controlled (NC) routers, lasers and water jets. Mark Goulthorpe of dECOi suggest that the calculation of space, form and structure will usurp design altogether and eclipse the archi-

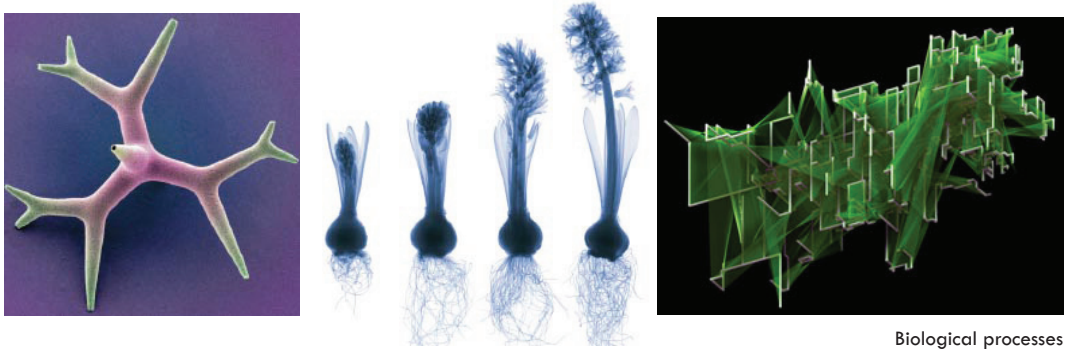
"Architectural form is not only an event of its internal, parametric-driven relational logic, but he also aims on answering to the dynamic, frequently variable, influences of its surroundings and socioeconomic context.

tect's previously **deterministic role**. What calculation challenges, he proposes, is "the very distinction of engineer, architect, etc. The separation of entities corresponding to the productive division of elements is precisely what is being called into question.

Computer

The computer, then, will no longer be merely a production, engineering or facilitation tool under the command of the architect-user but a generating entity with its own virtual intelligence or "knowledge" of the design process; the computer will function as a partner. Architecture is becoming a computational collaborative art based on the choreography of robotic manufacturing, while the architect, freed from the need to continuously invent anew, is becoming more like a choreographer of space and material production.

Computational models have been developed inspired by and out of fascination for evolutionary principles. By no means though are the computational model representative of evolution in nature but serve as metaphors for developing models of computation to deal with **the search** of large solution spaces. Evolutionary algorithms are essentially guided search mechanisms that use sophisticated model to inform the direction and area of search within the design space. The design space or solution space is the result of all possible combinations of the design parameters of the problem at hand and usually is far too large to be searched exhaustively.



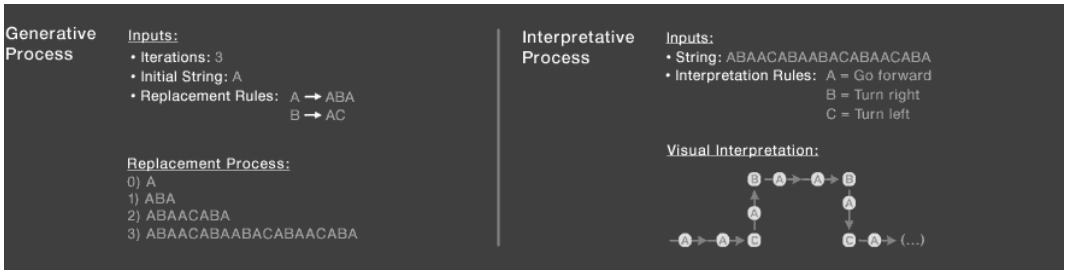
Biological processes

Evolution

Evolution drives the development of species in nature and is part of a very **complex system** of interaction between all elements. The compounds form out of the elements and the complex interactions on a chemical level lead to the formation of the first living beings. A

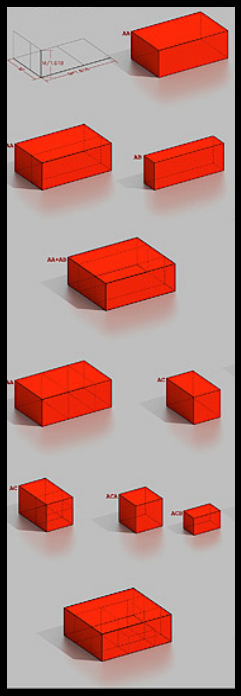
"Architecture is considered as a form of artificial life, subject, like the natural world, to principles of morphogenesis, genetic coding, replication and selection."

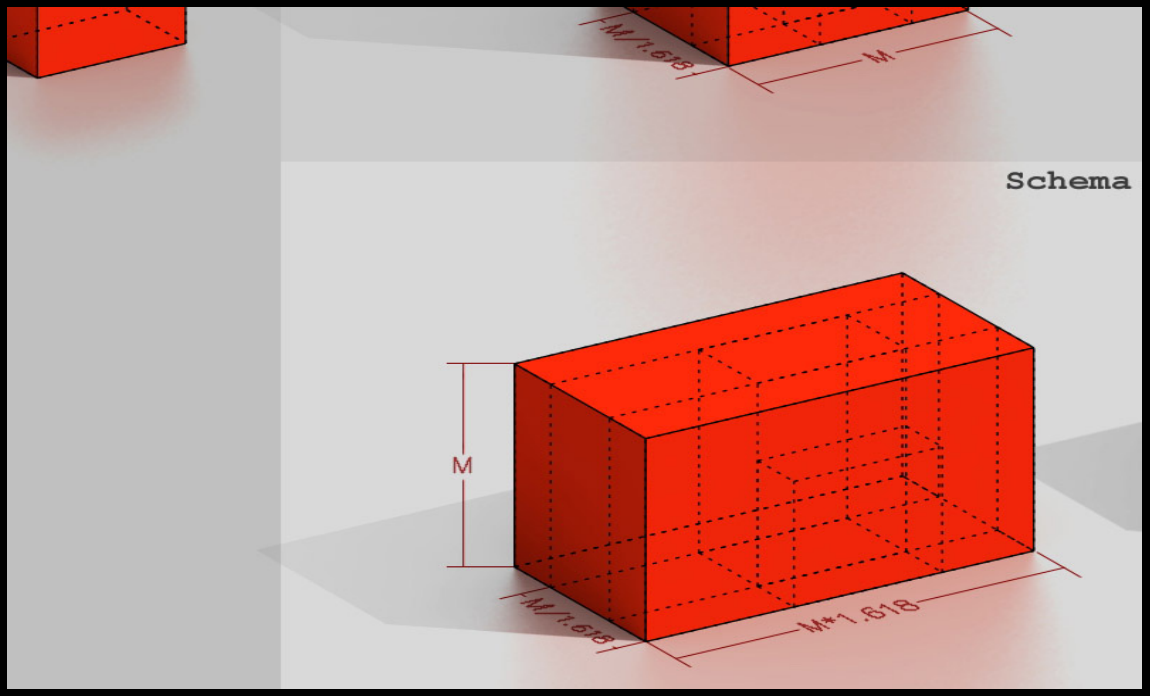
large part of evolution is the competition among entities in a more or less bounded environment. Competition creates selection and selection influences the proliferation of individuals who get to reproduce. Another core innovation that developed with evolution is the notion of **phenotype** and **genotype** which essentially means that the individual and its development is somewhat separated from the plan that is followed in the growth of the individual. This refinement makes it possible to carry over very complex traits from generation to generation and allows evolutionary principles to affect whole populations rather then just individuals.

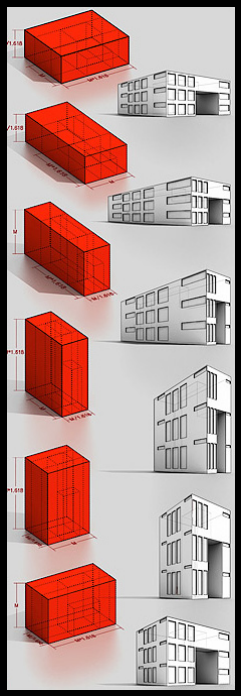


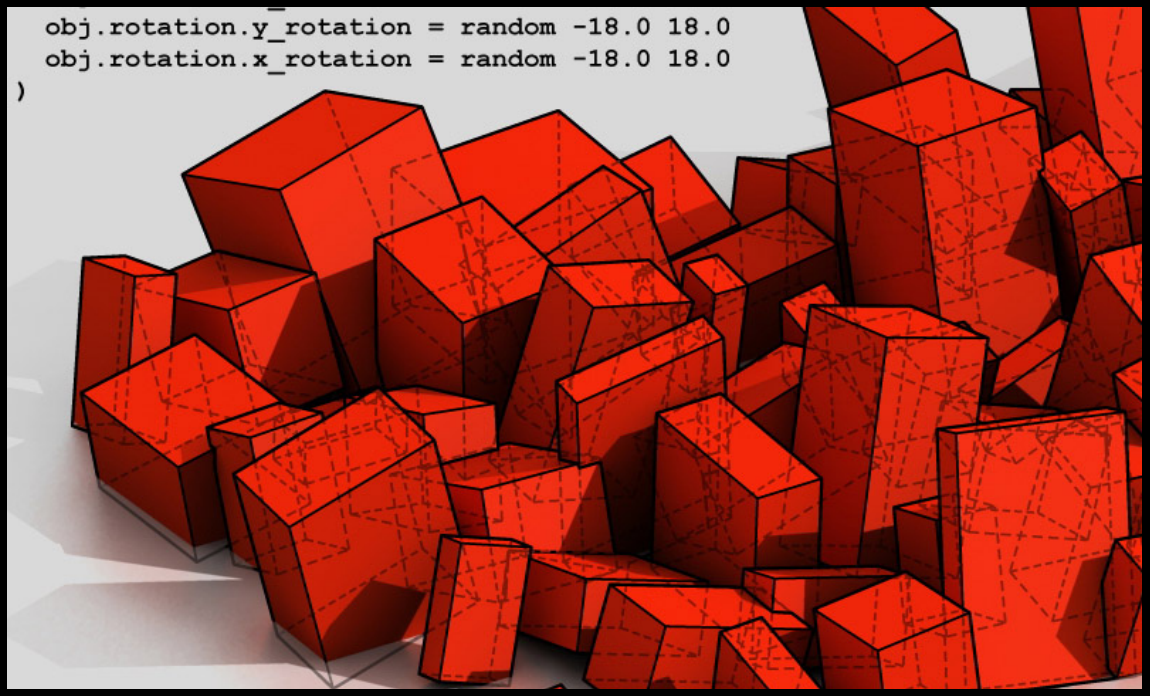
Growth process, L-systems

The use of generative principles as a form generator is very challenging, as the **fitness criteria** to test and evolve a form against can easily be too loose producing essentially random forms or to limiting which channels the results into the form that is described in the fitness function. As with any complex system the interactions and cross dependencies are very hard to foresee as the system grows large and requires a very fine balance of control and understanding of the system to be useful. The range of generative principles goes from optimization to almost random form generation and is as much a design task as anything else. The use of a generative principles does not justify any result that comes out of it but should be viewed as a design as anything else produced.




M/1.618
M
Schema
M
M/1.618
M*1.618




obj.rotation.y_rotation = random -18.0 18.0
obj.rotation.x_rotation = random -18.0 18.0
)

41

TECHNIQUE

SCRIPTS

Algorithm

Generative principles or other biological processes can be described according to the abstract logic and principles of algorithms. An algorithm is a process of addressing a problem in a finite number of steps. It is an articulation of either a strategic plan for solving a known problem or a stochastic search towards possible solutions to a partially known problem. In doing so, it serves as **a codification** of the problem through a series of finite, consistent, and rational steps. While most algorithms are designed with a specific solution in mind to a problem, there are some problems whose solution is unknown, vague, or ill-defined. In the latter case, algorithms become the means for exploring possible paths that may lead to potential solutions.

- DNA structure - Chromosome - Genetic Algorithms - Genetic Programming

Design Tooling; S. Dritsas, A. Kilian & Y. Loukissas

Theoretically, as long as a problem can be defined in logical terms, a solution may be produced that will address the problem's demands. An algorithm is a linguistic expression of the problem and as such it is composed of linguistic elements and operations arranged into spelling, and grammatically and syntactically correct statements. The linguistic articulation serves the purpose not only to describe the problem's steps but also to communicate the solution to another agent for further processing. In the world of computers, that agent is the computer itself. An algorithm can be seen as **a mediator** between the human mind and the computer's processing power. This ability of an algorithm to serve as a translator can be interpreted as bidirectional: either as means of dictating to the computer how to go about solving the problem, or as a reflection of a human thought into the form of an algorithm.

"An algorithm can be seen as a mediator between the human mind and the computer's processing power."

Interestingly, algorithms can generate other algorithms; not only precise, identical, multiple copies of themselves but also structured text (i.e. code) that when executed will behave as an algorithm. In fact, the process of composing an algorithm is also an algorithm in itself, that is. This self-referential property (which may be referred as **meta-algorithm**) is important in design for at least two reasons: first, like algorithms, design can be seen as a set of procedures that lead stochastically towards the accomplishment of a goal. Second, along the lines of homo faber homo fabricatus (i.e. we make a tool and the tool makes us), algorithms can be seen as design tools that lead towards the production of novel concepts, ideas, or forms, which, in turn, have an effect in the way designers think thereafter. That way of thinking is incorporated in the next generation of tools that will, in turn, affect the next generation of designers, and so on.

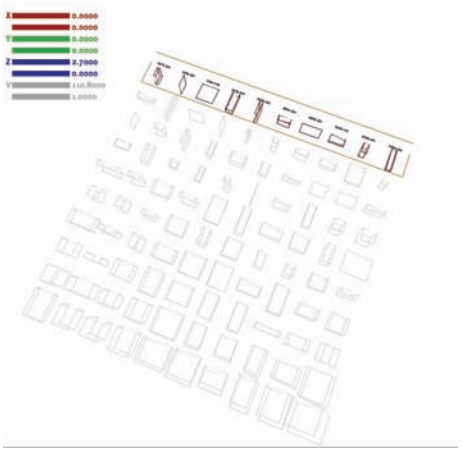
An generative algorithm is a principle that evolves populations of solutions to certain constrains, quite the same way evolutionary processes do in nature. Each population starts with a number of randomly generated individuals, which reproduce or not according to their performance against a 'fitness function'. The individual usually consist of a **genotype**, or certain encoded information on which the genetic algorithm operates (reproducing, mutating, recombining it), and the decoded and translated information, known as the **phenotype**, of which the fitness is evaluated. The system can then iteratively evolve fitter individuals.

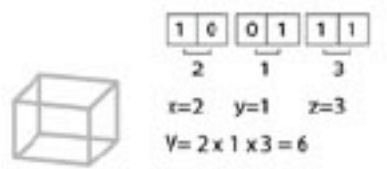
The following examples demonstrate a range of simple and more complex examples based on generative algorithms to demonstrate the principles and challenges of this computational approach. They are by no means exhaustive and any search in the cited literature or additional source will turn up a large number of additional examples in many fields besides architecture, which only relatively recently applied the idea of generative algorithms to design. These examples explores the qualities of design made by machines, devoid of any intention, assumptions or prejudices, and which often display a very peculiar form of mindlessly but relentlessly pounding against obstacles and problems until overcoming them.

"Algorithms can be seen as design tools that lead towards the production of novel concepts, ideas, or forms.

Most of the examples here are based on an array of **1**'s and **0**'s for the representation of the genome. The array is initiated by filling it with random values of one or zero at the beginning of each generation. Each individual is represented by one genome and the number of genomes represent the population. Usually the size of the population stays constant. Each individual in a generation is tested according to the set **fitness criteria**. Usually it is a weighted score derived from how well the individual performs a certain task or how close it fits a given goal. The genome is also referred to the genotype. The genotype is used as the instruction to produce the phenotype. The phenotype is the entity that is being tested for fitness. After each generation the genomes are ranked based on the performance of their phenotype and a percentage of the most successful is reinserted into the next random generation. This cycle continues for usually several thousands if not hundred thousands cycles in order to see if any trend in the **successful** genotypes emerges.

Ex.1: A generative algorithm is used to generate boxes of a certain volume given a choice of x-, y-, z-dimensions. The target values are weighted against each other which allows putting different emphasis on the values according to user preference. It is an example to illustrate a very simple case of multiple targets expressed through a weighted fitness function.

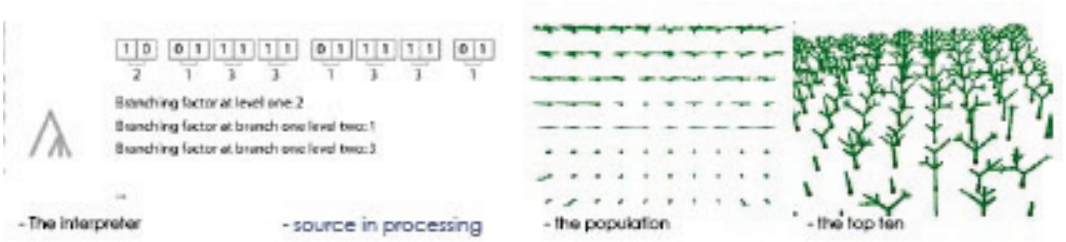




Design Tooling; S. Dritsas, A. Kilian & Y. Loukissas

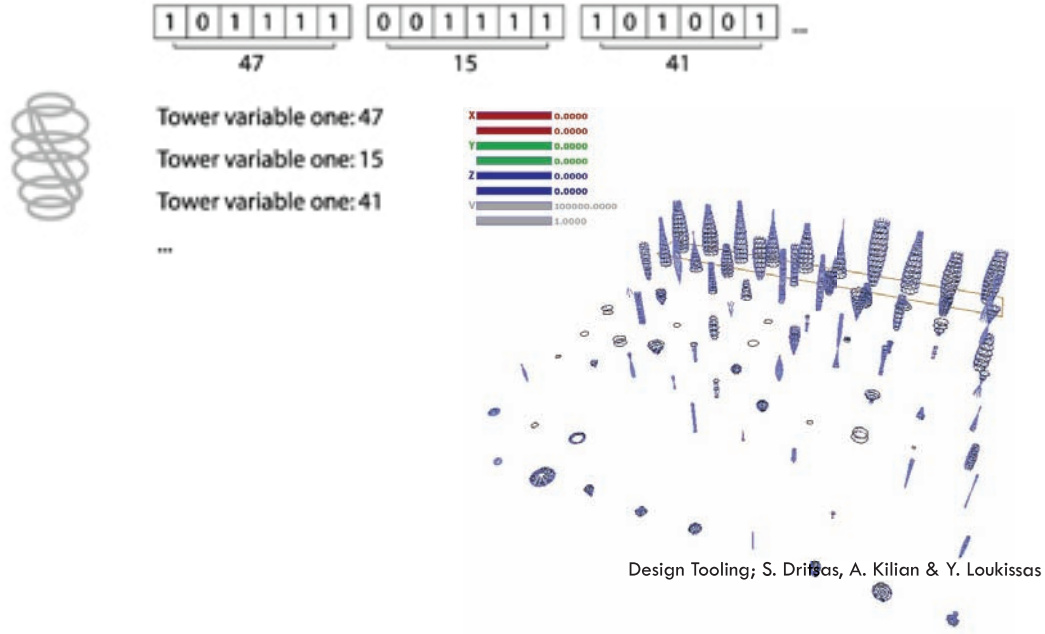
"Each individual in a generation is tested according to a set of fitness criteria's.

Ex.2: In the growth example (L-systems) the genome contains information that is being interpreted as the branching pattern during the growth of the tree. As an evaluation function the overall volume of the tree is measured against a target value.



Design Tooling; S. Dritsas, A. Kilian & Y. Loukissas

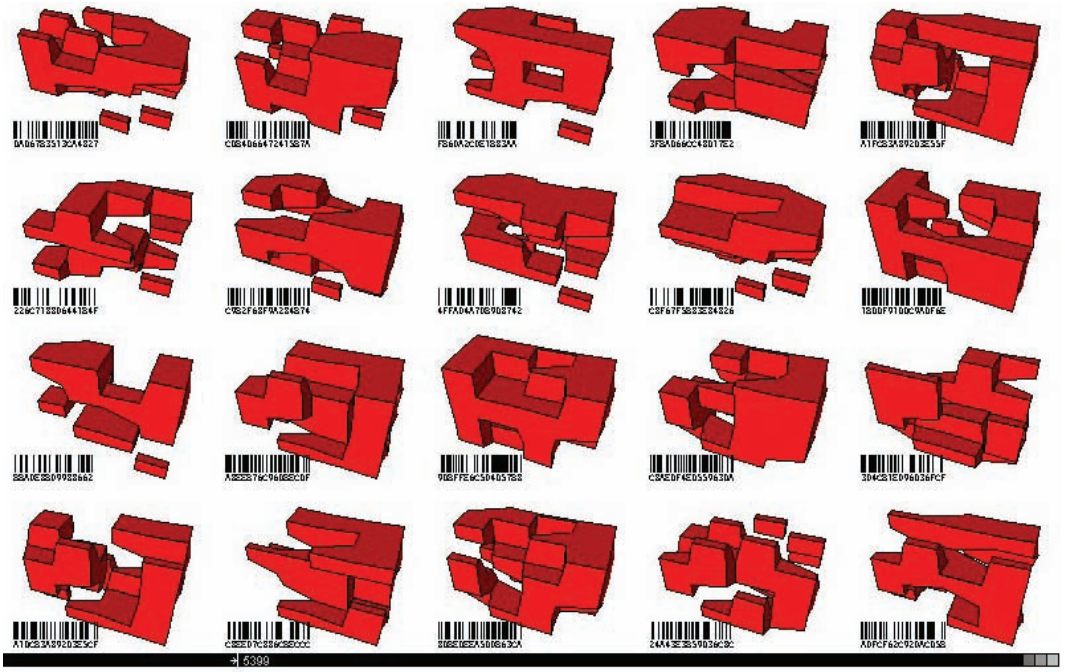
Ex.3: In this example a GA is used to generate tower variations based on the 6 control variables. A possible fitness criteria is volume, surface area and height. In the applet the towers are simply sorted by volume.



Design Tooling; S. Dritsas, A. Kilian & Y. Loukissas

"The individual usually consist of a genotype, or certain encoded information on which the genetic algorithm operates and the decoded and translated information, known as the phenotype, of which the fitness is evaluated.

Ex.4: ArchiKluge: The genetic algorithm takes two randomly selected individuals and substitutes the least fitted by their common offspring (produced by a cross over operator). Mutations are also randomly performed in the individuals. The fitness function consist of the addition of each cell's added 'shortest paths', a measure often used in network analysis (and in space analysis such as Bill Hillier's Space Syntax). This means that for each cell in the 4x4x4 lattice, the shortest path to every other cell is calculated. This Fitness function is intended only as an example, and it could be interesting in the future to incorporate an interface that allows user defined fitness functions. For illustrating the resulting circulations through the evolved layouts, the paths left by random walkers, or agents that move randomly through the lattice have been used.



The black and white stripes are the bits (1 or 0s) of the genotype , in hexadecimal base under them. The arrow and numbers at the bottom show the progress of the fitnesses. Drag the mouse to rotate the individual phenotypes. Use the buttons to stop and reset the evolution, or change between displays, solid or circulation paths.

Archikluge, Pablo Miranda Carranza

"A Genetic Algorithm is a program that evolves populations of solutions to certain constrains, quite the same way evolutionary processes do in nature.

In design, algorithms can be used to solve, organize or explore problems with increased visual or organizational complexity. In its simplest form, a computational algorithm uses **numerical methods** to address problems. Numbers are often regarded as discrete quantitive units that are utilized for measuring, yet in computational terms numbers can be constructed that can address an infinite degree of division thus exhibiting theoretical continuity. The basic linguistic elements used in algorithms are constants, variables, procedures, classes, and libraries and the basic operations are arithmetical, logical, combinational, relational, and classificatory arranged under specific grammatical and syntactical rules. These elements and operations are designed to address the numerical nature of computers while at the same time provide the means for composing logical patterns. For example, pixels on a computer screen are numerical elements defined by their colour value and their x and y coordinates, but at the same time they can be addressed as visual compositions defined by logical patterns.

In this part it is made clear, by means of an example script, how previous algorithms look like if they are formulated to **a syntactic code** which can be understand by the computer. Initially it was the intention offering some fundamental linguistic elements from which one could build an algorithm as a designer. However this would the same thing as writing a guide for a particular script language that it is bound of course to a specific application such as Maya, Rhino, Max, etc. Such guides already exist, are very extended and available in each software application. As a designer, an example script and the visualized generations would offer a better insight of the applied syntax. Several rules and operations would come to light and thus the designer can stimulate himself to extend his knowledge.

Ex.

For these reasons and with an eye on my final project (`science park' at Waterschei, Genk, Belgium), where I will try to divide a three-dimensional structure into lots according to defined principles translated in a script, I have chosen to present, `Algoritmic Architecture', a project of Daniel Davis as an example. Also because I have learnt to work with 3d Studio Max at school and Daniel Davis has written his project 'Algorithm Architecture' in MAXscript. The project was a 3rd year design project, for Victoria School of Architecture in New Zealand. The project was to design a hotel, on a set site, with a set of functions (eg. a reception area, sleeping areas, restaurant ect.)

"In design, algorithms can be used to solve, organize or explore problems with increased visual or organizational complexity.

Approach

Daniel Davis was getting really frustrated with the design pedagogy he was being taught at school. It was essentially a post-modern one, where metaphor became the guiding principal to design. We were taught so start with 'a concept' (earthquakes, trasiant nature of space, rivers) and use this as a metaphor to guide the design of a building. He did not agree with this process as it seemed to have nothing to do with the actual in habitation, function of architecture. Humans have been designing by chance, taking a guess, seeing if it works and passing very little of the information on the chances of getting it right a low. Architects like to think of themselves as gods though; infinity aware of the consequences of there decisions. To be a god you must be infinitely aware you have to be infinitely complex to get it right the first time. Humans are far from it. Daniel Davis would then conclude that algorithmic architecture coupled with the rise of computational power is a way to improve design. His project set out to prove this. He says that it is important with computer games like Spore coming out, that architects take control of these methods before computer programmers do…

Algorithmic Hotel, Daniël Davis

"Architects should take control of these evolutionary design methods before computer programmers do..."

His project was written in Maxscript. Maxscript is an easy language, is intended mainly for 3d forms. The script for his project has 3 main functions:

1. The first is to translate a digital chromosomes into built form. This seems easy but it is really hard. How do you express a building as a string of numbers?

2. The second is to evaluate the chromosomes (find how successful they are). The more complex evaluations were based on room sizes, function and adjacency. It was quite primitive but if there was a framework, it would not take long for it to get very complex.

3. The third function was to breed the chosen chromosomes to make children of them. The challenge was to get the right rate of evolution so that there was enough diversity in the population that new solutions were being found, but enough cohesiveness that the population had traits. Unfortunately he did not get this right. He found out by running the evaluation over the generations and the mean did not improve. But he did not have time to fix it.


```
function startData = (
--sets the start data*/
    --access, room[room number][
    --1 = room name
    --2 = array of x dimension
    --3 = array of y dimension
    --4 = array of area
    --5 = aray z dimension
    --6 = array of values
    --7 = array of joins
    --8 = array of optional joins, false if doesn't]
    --9 = human name of room
    --10 = colour of room?
    --11 = array of levels it can go on, in order of preferance
    global ROOM = #();
```

"Scripts are designed to address the numerical nature of computers while at the same time provide the means for composing logical patterns.

```
    --reception
    ROOM[1] = #(“R1”, #(4.0, 6, 10), #(4.0, 6, 10), #(15, 30, 100), #(1, 2, 4), #(2, 4, 4), #(“R2”, “R3”, “X1”), #(“X4”, “O4”), “Wainting_Area”, #(100, 100, 100), 1);
    ROOM[2] = #(“R2”, #(2.0, 4, 6), #(2.0, 4, 6), #(6, 12, 20), #(1, 2, 2), #(1, 4, 3), #(“R1”, “B1”), false, “CheckIn”, #(100, 100, 100), 1);
    ROOM[3] = #(“R3”, #(3.2, 3.2, 5), #(1.8, 1.8, 4), #(5.7, 5.7, 10), #(1, 1, 1), #(1, 1, 1), #(“R1”), false, “Toilets”, #(100, 100, 100), 1);

    --back of house
    ROOM[4] = #(“B1”, #(4.0, 6.0, 10.0), #(4.0, 6.0, 10.0), #(20.0, 30, 40), #(1, 1, 1), #(1, 5, 2), #(“R2”, “B2”, “B4”, “B5”, “B6”, “B7”), false, “Meeting_Room”, #(200, 0, 0), 5);
    ROOM[5] = #(“B2”, #(4.0, 6.0, 10.0), #(4.0, 6.0, 10.0), #(20.0, 30, 40), #(1, 1, 1), #(2, 4, 2), #(“B1”, “B3”), false, “Staff_room”, #(200, 0, 0), 5);
    ROOM[6] = #(“B3”, #(0.4, 0.5, 0.7), #(12.0, 16.0, 20.0), #(8.0, 8.0, 8.0), #(1, 1, 1), #(1, 1, 1), #(“B2”), false, “Staff_lockers”, #(200, 0, 0), 5);
    ROOM[7] = #(“B4”, #(2.5, 3.0, 5.0), #(2.5, 4.0, 5.0), #(8.0, 10.0, 20.0), #(1, 1, 1), #(3, 5, 3), #(“B1”), false, “Assistant”, #(200, 0, 0), 5);
    ROOM[8] = #(“B5”, #(2.5, 3.0, 5.0), #(2.5, 4.0, 5.0), #(8.0, 10.0, 20.0), #(1, 1, 1), #(3, 5, 3), #(“B1”), false, “Accounting”, #(200, 0, 0), 5);
    ROOM[9] = #(“B6”, #(3.0, 4.0, 5.0), #(3.0, 5.0, 5.0), #(8.0, 14.0, 20.0), #(1, 1, 1), #(3, 5, 3), #(“B1”), false, “Manager”, #(200, 0, 0), 5);
    ROOM[10] = #(“B7”, #(1.0, 1.0, 2.0), #(1.6, 1.6, 2.0), #(1.6, 1.6, 4.0), #(1, 1, 1), #(1, 1, 1), #(“B1”), false, “Toilets”, #(200, 0, 0), 5);

    --Circulation
    ROOM[11] = #(“X1”, #(1.5, 2.0, 3.0), #(1.5, 2.0, 3.0), #(3.0, 4.0, 7.0), #(5, 5, 5), #(1, 1, 1), #(“C1”, “R1”), false, “Lift”, #(0, 200, 200), 0);
    ROOM[18] = #(“X2”, #(2.0, 2.5, 3.0), #(2.0, 2.5, 3.0), #(4.0, 6.0, 9.0), #(5, 5, 5), #(1, 1, 1), #(“Y2”), false, “Service_Lift”, #(0, 200, 200), 0);
    --ROOM[22] = #(“X3”, #(3.0, 4.0, 5.0), #(3.0, 4.0, 5.0), #(12, 12, 12), #(5, 5, 5), #(1, 1, 1), #(“X4”, “X5”, “X6”, “X7”), , “Fire_Exit”, #(0, 200, 200), 0);

    --Outside
    ROOM[12] = #(“O4”, #(0.0, 2.0, 5.0), #(0.0, 2.0, 5.0), #(0, 2.0, 20.0), #(1, 1, 1), #(1, 1, 1), #(“Z1, Z2”, “X1”), #(“X1”, “R1”), “Common_Area”, #(0, 200, 0), 6);

    --cafe
    ROOM[13] = #(“Y1”, #(5.0, 9.0, 15.0), #(5.0, 9.0, 15.0), #(60.0, 80.0, 100.0), #(1, 1, 3), #(4, 4, 4), #(“Z2”, “O4”, “O5”), false, “Seating”, #(0, 200, 0), 6);
    ROOM[14] = #(“Y3”, #(2.0, 3.0, 4.0), #(4.0, 5.0, 6.0), #(8.0, 12.0, 20.0), #(1, 1, 2), #(2, 4, 1), #(“Y1”, “Y2”), false, “Buffet_Table”, #(0, 200, 0), 6);
    ROOM[15] = #(“Y2”, #(4.0, 6.0, 9.0), #(4.0, 7.0, 9.0), #(40.0, 40.0, 60.0), #(1, 1, 1), #(1, 1, 1), #(“Z1”, “Y3”, “X2”), false, “Kitchen”, #(0, 200, 0), 6);

    --Bar
    ROOM[16] = #(“Z2”, #(10.0, 13.0, 20.0), #(10.0, 13.0, 20.0), #(150.0, 200.0, 250.0), #(1, 1, 3), #(4, 3, 5), #(“Z1”, “Y1”, “O4”, “O5”), false, “Seating_Bar”, #(0, 200, 100), 6);
    ROOM[17] = #(“Z1”, #(2.0, 2.50, 4.0), #(6.0, 10.0, 12.0), #(17.0, 20.0, 50.0), #(1, 1, 2), #(1, 3, 3), #(“Z2”, “Y2”), false, “Bar”, #(0, 200, 100), 6);

    --service Areas
    ROOM[19] = #(“S2”, #(0.0, 3.0, 5.0), #(0.0, 3.0, 5.0), #(0.0, 9, 25), #(1, 1, 1), #(1, 1, 1), #(“S1”, “X2”), false, “Access”, #(200, 50, 100), 7);
    ROOM[20] = #(“S1”, #(7.0, 8.5, 20.0), #(7.0, 8.5, 20.0), #(50.0, 100, 120), #(1, 1, 1), #(1, 1, 1), #(“S2”), false, “Storage”, #(200, 50, 100), 7);

    --carparking
    ROOM[21] = #(“C1”, #(10.0, 30.0, 30.0), #(10.0, 20.0, 30.0), #(400.0, 1200.0, 1400.0), #(1, 1, 1), #(1, 1, 1), #(“X1”, “X2”), false, “CarPark”, #(0, 0, 0), 0);

    --corridor’s
```

```
    ROOM[22] = #("X4A", #(0.0, 2.0, 20.0), #(2.0, 2.5, 3.0), #(0.0, 30, 90), #(1, 1, 1), #(1, 1, 1), #("X1", "X2"), false,
"Corridor", #(250, 250, 250), 4);
    ROOM[23] = #("X4B", #(2.0, 2.5, 3.0), #(0.0, 2.0, 20.0), #(0.0, 30, 90), #(1, 1, 1), #(1, 1, 1), #("X4A"), false,
"Corridor", #(250, 250, 250), 4);
    ROOM[24] = #("X5A", #(2.0, 2.5, 3.0), #(0.0, 2.0, 20.0), #(0.0, 30, 90), #(1, 1, 1), #(1, 1, 1), #("X1", "X2"), false,
"Corridor", #(250, 250, 250), 3);
    ROOM[25] = #("X5B", #(0.0, 2.0, 20.0), #(2.0, 2.5, 3.0), #(0.0, 30, 90), #(1, 1, 1), #(1, 1, 1), #("X5A"), false,
"Corridor", #(250, 250, 250), 3);
    ROOM[26] = #("X6A", #(0.0, 2.0, 20.0), #(2.0, 2.5, 3.0), #(0.0, 30, 90), #(1, 1, 1), #(1, 1, 1), #("X1", "X2"), false,
"Corridor", #(250, 250, 250), 2);
    ROOM[27] = #("X6B", #(2.0, 2.5, 3.0), #(0.0, 2.0, 20.0), #(0.0, 30, 90), #(1, 1, 1), #(1, 1, 1), #("X6A"), false,
"Corridor", #(250, 250, 250), 2);

    ROOM[27] = #("room1", #(3.0, 5, 10), #(3.0, 5, 10), #(10, 25, 50), #(1, 1, 1), #(1, 1, 1), #("X6A"), false, "Room",
#(200, 200, 200), 2);
    ROOM[28] = #("room2", #(3.0, 5, 10), #(3.0, 5, 10), #(10, 25, 50), #(1, 1, 1), #(1, 1, 1), #("X6A"), false, "Room",
#(200, 200, 200), 2);
    ROOM[29] = #("room3", #(3.0, 5, 10), #(3.0, 5, 10), #(10, 25, 50), #(1, 1, 1), #(1, 1, 1), #("X6A"), false, "Room",
#(200, 200, 200), 2);
    ROOM[30] = #("room4", #(3.0, 5, 10), #(3.0, 5, 10), #(10, 25, 50), #(1, 1, 1), #(1, 1, 1), #("X6A"), false, "Room",
#(200, 200, 200), 2);

    ROOM[31] = #("room5", #(3.0, 5, 10), #(3.0, 5, 10), #(10, 25, 50), #(1, 1, 1), #(1, 1, 1), #("X4A"), false, "Room",
#(200, 200, 200), 2);
    ROOM[32] = #("room6", #(3.0, 5, 10), #(3.0, 5, 10), #(10, 25, 50), #(1, 1, 1), #(1, 1, 1), #("X4A"), false, "Room",
#(200, 200, 200), 2);
    ROOM[33] = #("room7", #(3.0, 5, 10), #(3.0, 5, 10), #(10, 25, 50), #(1, 1, 1), #(1, 1, 1), #("X4A"), false, "Room",
#(200, 200, 200), 2);
    ROOM[34] = #("room8", #(3.0, 5, 10), #(3.0, 5, 10), #(10, 25, 50), #(1, 1, 1), #(1, 1, 1), #("X4A"), false, "Room",
#(200, 200, 200), 2);

    ROOM[35] = #("room9", #(3.0, 5, 10), #(3.0, 5, 10), #(10, 25, 50), #(1, 1, 1), #(1, 1, 1), #("X5A"), false, "Room",
#(200, 200, 200), 3);
    ROOM[36] = #("room10", #(3.0, 5, 10), #(3.0, 5, 10), #(10, 25, 50), #(1, 1, 1), #(1, 1, 1), #("X5A"), false, "Room",
#(200, 200, 200), 3);
    ROOM[37] = #("room11", #(3.0, 5, 10), #(3.0, 5, 10), #(10, 25, 50), #(1, 1, 1), #(1, 1, 1), #("X5A"), false, "Room",
#(200, 200, 200), 3);
    ROOM[38] = #("room12", #(3.0, 5, 10), #(3.0, 5, 10), #(10, 25, 50), #(1, 1, 1), #(1, 1, 1), #("X5A"), false, "Room",
#(200, 200, 200), 3);

    ROOM[39] = #("room13", #(3.0, 5, 10), #(3.0, 5, 10), #(10, 25, 50), #(1, 1, 1), #(1, 1, 1), #("X5B"), false, "Room",
#(200, 200, 200), 3);
    ROOM[40] = #("room14", #(3.0, 5, 10), #(3.0, 5, 10), #(10, 25, 50), #(1, 1, 1), #(1, 1, 1), #("X5B"), false, "Room",
#(200, 200, 200), 3);
    ROOM[41] = #("room15", #(3.0, 5, 10), #(3.0, 5, 10), #(10, 25, 50), #(1, 1, 1), #(1, 1, 1), #("X5B"), false, "Room",
#(200, 200, 200), 3);
    ROOM[42] = #("room16", #(3.0, 5, 10), #(3.0, 5, 10), #(10, 25, 50), #(1, 1, 1), #(1, 1, 1), #("X5B"), false, "Room",
#(200, 200, 200), 3);

    --array of xy points making the site bounds
    global SITE = #(#(0.0,0.0), #(20.0, 0.0), #(20.0, 32.5), #(0, 32.5));

    --gives an approximage site square
    global SITEAPPROX = #(20.0, 32.5)

    --sets genome
    global DNA = #();

    --sets the loop safty
    SAFE = 20;

    --Floor height
```

```
    global FLOORHEIGHT = 2.7;

    --Minimum joint between related rooms
    global DOORWAY = 1;
)

--#############################################################
--box creation and movement
--#############################################################

function moveBox name xLoc yLoc zLoc = (
--moves box
    xLoc = xLoc as string
    yLoc = yLoc as string
    zLoc  = FLOORHEIGHT * zLoc;
    zLoc = zLoc as string

    --moves the box
    str = "$" + name + ".pos = [" + xLoc + ", " + yLoc + ", " + zLoc + "]"
    execute str
)

function sizeBox name width height length = (
--resizes box*/
    width = width as string
    height = height as string
    length = length as string

    --resizes the box
    str = "$" + name + ".width = " + width
    ex      str
    st  $" + name + ".height = " + height
    ex      str
    str = "$" + name + ".length = " + length
    execute str
)

function createBox width height length xLoc yLoc zLoc cName colour  = (
--creates a box, clours it, names it, resizes it, moves it
    mybox = box length:length width:width height:height
    mybox.wireColor = (color colour[1] colour[2] colour[3])
    mybox.name = cName

    sizeBox cName width height length

    moveBox cName xLoc yLoc zLoc
)

function createPlace parent width length side percent d = (
    --creates the xy location of a room,
    --where d is the dna array to use
    --based on its parent, the side it attaches to, the percent along it is
    --returns array of xy coordinates

    --sets the x, y position to its parent
    x = DNA[d][parent][5];
    y = DNA[d][parent][6];

    --finds how far to move it relative to its parent
    if(side == 1) then (/*top*/
        y = y + (length + DNA[d][parent][3]) / 2;
        --parents position + half of the parents length and half of the length
```

```
        x = x + (width + DNA[d][parent][2] - DOORWAY) * percent / 100 - (width + DNA[1][parent][2]) / 2
        --parents position + percentage of width and parents width - half of width and parents width
    )else if(side == 2) then (/*left*/
        x =  x + (width + DNA[d][parent][2]) / 2;
        y = y + (length + DNA[d][parent][3] - DOORWAY) * percent / 100 - (length + DNA[1][parent][3]) / 2
    )else if(side == 3) then (/*bottom*/
        y = y - (length + DNA[d][parent][3]) / 2;
        x = x + (width + DNA[d][parent][2] - DOORWAY) * percent / 100 - (width + DNA[1][parent][2]) / 2
    )else if(side == 4) then (/*right*/
        x = x - (width + DNA[d][parent][2]) / 2;
        y = y + (length + DNA[d][parent][3] - DOORWAY) * percent / 100 - (length + DNA[1][parent][3]) / 2
    )
    return #(x, y);
)

function createFromDNA number addOn addOny= (
--Crates rooms from the DNA given to it, where number is the DNA strand and addOn is the amount it is offcentered on the x
axis
    for i = 1 to DNA[number].count do (--loops through all of the rooms
        --creates the box for that peice of dna
        createBox DNA[number][i][2] DNA[number][i][4] DNA[number][i][3] (DNA[number][i][5] + addOn) (DNA[number][i][6] +
addOny) DNA[number][i][7] (number as string + addOny as string + DNA[number][i][1]) DNA[number][i][9];
    )
)

--#############################################################
--room evaluation
--#############################################################

function isIn x y width length  = (
--checks that a box is inside the boundries
--returns true if inside, returns false if not
        if (x < width/2) then (
            return false;
        )else if (x > SITEAPPROX[1] - width/2) then (
            return false;
        )else if (y < length/2) then (
            return false;
        )else if (y > SITEAPPROX[2] - length/2) then (
            return false;
        ) else (
            return true;
        )
)

function isAlone x y z width length height d= (
--z as level and height as meters
--checks that a box is not touching other boxes
--returns true if not touching, returns false if touching
    for i = 1 to DNA[d].count do (--loops through the existing rooms
        -- checks to see if it is on the same level
        --if it is not, then it skips this loop
        if(isInterceting x y z width length height DNA[d][i][5] DNA[d][i][6] DNA[d][i][7] DNA[d][i][2] DNA[d][i][3] DNA[d]
[i][4]) then (
            return false;
        )
    )
    --looped though all rooms, none touch.
    return true;
)

function isInterceting x1 y1 z1 w1 l1 h1 x2 y2 z2 w2 l2 h2 = (
```

```
--takes the 12 peices of date and finds if the cubes intercept.
--returns true if they intercet, returns false if they dont.
--can make this faster
    diffZ = z2 - z1;
    if (diffZ >= h1 / FLOORHEIGHT) then (
        return false;--too high
    )
    if(diffZ + h2 / FLOORHEIGHT <= 0) then (
        return false;--below
    )
    --if box is on same level, checks to see if there is overlap.
    diffX = x1 - x2;
    if (diffX < (w2 / 2 + w1 / 2) and diffX > (-w2 / 2 - w1 / 2)) then (
    --gets the x difference of the room
        diffY = y1 - y2;
        if (diffY < (l2 / 2 + l1/ 2) and diffY > (-l2 / 2 - l1 / 2)) then (
        --gets the y difference of the room
            return true;
        )
    )
    return false;
)

function isTouching x1 y1 w1 l1 x2 y2 w2 l2 = (
-- returns true if touching
--retuns false is not touching
    diffX = x1 - x2;
    diffY = y1 - y2;
    if((diffX == w1/2 + w2 /2) or (diffX == -w1/2 - w2 / 2)) then (
        if (diffY < (l2 / 2 + l1/ 2) and diffY > (-l2 / 2 - l1 / 2)) then (
            return true;
        )
    )
    if((diffY == l1/2 + l2 /2) or (diffY == -l1/2 - l2 / 2)) then (
        if (diffX < (w1 / 2 + w2/ 2) and diffX > (-w2 / 2 - w1 / 2)) then (
            return true;
        )
    )
    return false;
)

function isSuccess x y z width length d group n= (
--checks if created box is on right level for parent and touching parent
--if it is touching returns true, if not returns false.
    --checks that it is on right level
    if (n == 1) then (--it is the first box, no parent to check for.
        return true;
    )
    parent = whoIsParent n;
    zPair = getLevel group parent d;
    if (zPair[2] == false) then ( --then must be on exact level
        if (zPair[1] != z) then ( --checks that it is on exact level
            return false; --if not returns false
        )
    )
    --is on right level
    if(isTouching x y width length DNA[d][parent][5] DNA[d][parent][6] DNA[d][parent][2] DNA[d][parent][3]) then (
        return true; --on right level and touching
    ) else (
        return false; --on right level but not touching
    )
```

```
)

function whoIsParent number = (
--gives the location in the array of this rooms first parent
    for i = 1 to ROOM[number][7].count do (--cycles through all possible parents
        for k = 1 to ROOM.count do (--cycles through all the rooms
            if (ROOM[number][7][i] == ROOM[k][1]) then (--checks to see if room matches the parent
                if (k < number) then (
                    return k;--returns and ends if it does
                )
            )
        )
    )
    print "###error in the whoIsParent, no parent found###";
    return false;
)

function getLevel group parent d = (
--number = room number, parent = parent, d=dna.
--either returns array, [1] = level group is on or creates a level for the group
--[2] = false if cannot go on any other floor.
    --if the number is set to 0 - 4 then build on that floor
    if (group <= 4) then (
        return #(group, false);
    )
    --if not look to see if any other room in the group has been made
    for k = 1 to DNA[d].count do (--cycles through all the DNA loops
        if (DNA[d][k][10] == group) then (--if it has been made
            --return the height for that group
            return #(DNA[d][k][7], false);
        )
    )
    --if not then set the height for the group;
    return #(DNA[d][parent][7] + (random 0 ((DNA[d][parent][4]-1) / FLOORHEIGHT)), true);
)

function rateSize width length number = (
--checks to see if it is too big or too small.
--returns a rating from 0 + where 0 is the best.
    rate = 0;
    area = width * length;
    if (width < ROOM[number][2][1] or width > ROOM[number][2][3]) then rate += 1;
    if (length < ROOM[number][3][1] or length > ROOM[number][3][3]) then rate += 1;
    if (area < ROOM[number][4][1] or area > ROOM[number][4][3]) then rate += 1;
    return rate;
)

--#############################################################
--saving and reading
--#############################################################
function saveDNA dstart dend knowPath= (
--saves a strand of dna to file.
    if knowPath == false then (
        output_name = getSaveFileName caption:"SpeedSheet File" types:"Text (*.txt)|*.txt|All Files (*.*)|*.*|"
        if output_name != undefined then (
                output_file = createfile output_name
                for d = dstart to dend do (
                    for i = 1 to DNA[d].count do (
                        format "%\n" DNA[d][i] to:output_file
                    )
                    format "####\n" to:output_file
                )
```

```
                close output_file
                edit output_name
        )--end if
    )else (
        output_name = ((GetDir #export)+"/out_" + (knowPath as string) + ".txt")
        if output_name != undefined then (
                output_file = createfile output_name
                for d = dstart to dend do (
                    for i = 1 to DNA[d].count do (
                        format "%\n" DNA[d][i] to:output_file
                    )
                    format "####\n" to:output_file
                )
                close output_file
                edit output_name
        )--end if
    )
)

function openDNA op= (
    DNA = #();
    d = 1;
    n = 1;
    create = true;
    in_name = getOpenFileName()
    --in_name = ((GetDir #export)+"/out_" + (op as string) + ".txt")
    if in_name != undefined then(
        in_file = openFile in_name
        if in_file != undefined then (
            while eof in_file == false do (
                if create == true then (
                    DNA[d] = #();
                    create = false;
                )
                num_verts = readLine in_file
                if num_verts != "####" then (
                    DNA[d][n] = execute(num_verts);
                    n += 1;
                ) else (
                    d += 1;
                    create = true;
                    n = 1;
                )
            )
            close in_file
        )
    )
)

--#############################################################
--building creation
--#############################################################

function createRoom n d width: length: height: side: percent: = (
    --creates random room of n, from the ROOM array
    --adds it to d in the dna array.
    nameHuman = ROOM[n][9];
    nameReal = ROOM[n][1];
    colour = ROOM[n][10];
    if width == unsupplied then width = random ROOM[n][2][1] ROOM[n][2][3];
    if length == unsupplied then length = random ROOM[n][3][1] ROOM[n][3][3];
    if height == unsupplied then height = random ROOM[n][5][1] ROOM[n][5][3] * FLOORHEIGHT;
```

```
group = ROOM[n][11];
--chooses side and sets percentage to move
if side == unsupplied then side = random 1 4;
if percent == unsupplied then percent = random 1 100;
x=0;
y=0;
z=0;
--chooses who is the parent of the object
parent = whoIsParent n;
zPair = getLevel group parent d;
z = zPair[1];
good = false;

if(n == 1) then (
    --gets Z hieght for floor
    zPair = getLevel n parent d;
    z = zPair[1];
--if the first one, lines up on the front facade
    y = length / 2
    x = (SITEAPPROX[1] - width) * percent / 100 + width / 2
)

if(n > 1) then(--if not the first box
    for k = 1 to 4 do ( --loops through different heights until one is found to fit
        for j = 1 to 40 do ( --loops through different sizes until one is found to fit
            for i = 1 to 40 do ( --loops through different positions until one is found to fit

                --calculates where a box goes based on the side it attaches to and the percentage on it
                pair = createPlace parent width length side percent d;
                x = pair[1];
                y = pair[2];

                --check that the box is inbounds, and not touching anything
                --if not in, set goood to false which keeps loop searching for a good place
                if (isIn x y width length) then (
                    if (isAlone x y z width length height d) then (
                        good = true;
                        exit;
                    )
                )

                --create gives a new side to try
                side = random 1 4;
                percent = random 1 100;

            --###################
            )--end for, position

            --check that the box is inbounds
            if (isIn x y width length) then (
                if (isAlone x y z width length height d) then (
                    good = true;
                    exit;
                )
            )
            --gives it new width and length to try if doesn't fit.
            width = random ROOM[n][2][1] ROOM[n][2][3];
            length = random ROOM[n][3][1] ROOM[n][3][3];
        --###################
        )--end for, sizes

        if (isIn x y width length) then (
```

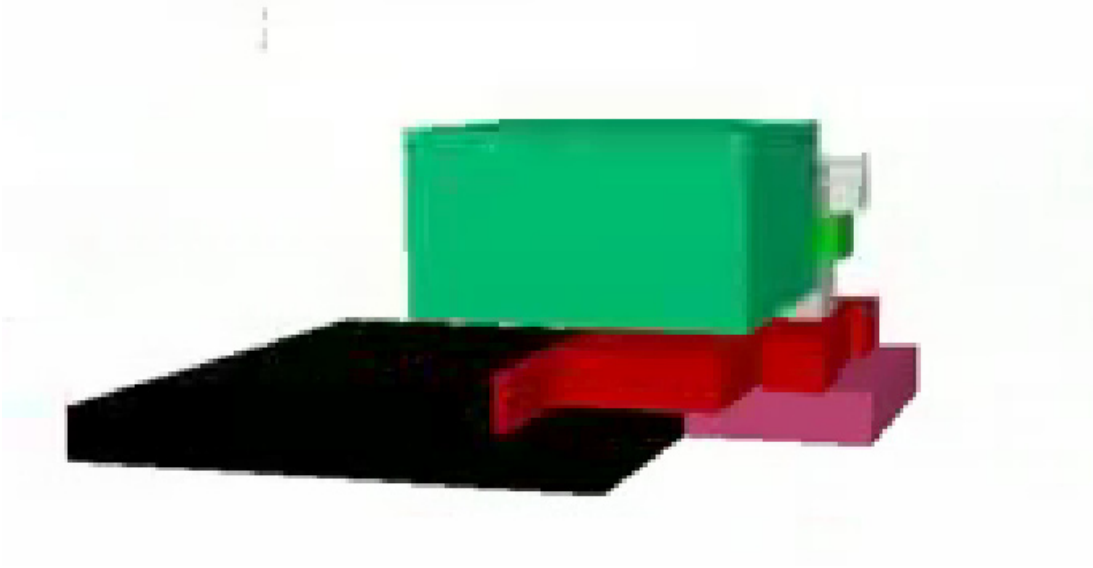
```
                if (isAlone x y z width length height d) then (
                    good = true;
                    exit;
                )
            )
            if (zPair[2] == false) then (--there is no where else for it to go
                exit;
            )
            --gives new height to test
            zPair = getLevel group parent d;
            z = zPair[1];
        )--end for, height test
    )--end if

    --createBox width height length x y z nameHuman colour;
    --if (good == false) then (
        --if (nameReal == "room") then (
            --continue;
        --)
    --)
    if (n <= 27) then (
        DNA[d][n] = #(nameReal, width, length, height, x, y, z, nameHuman, colour, group, side, percent, n);
    )else (
        if ((isIn x y width length) and (isAlone x y z width length height d)) then (
            DNA[d][n] = #(nameReal, width, length, height, x, y, z, nameHuman, colour, group, side, percent, n);
        ) else (
            DNA[d][n] = #(nameReal, 0, 0, 0, 0, 0, 0, nameHuman, colour, group, side, percent, n);
        )
    )
)

--############################################################
--DNA functions
--############################################################

function evaluate d = (
--evaluates a the given peice of dna and returns its rating as a number, smaller is better.
--N check how big the room is.
--N check if the room touches all it needs to.
    --sees how many times the room is outside the bounds and how many times they touch.
    roomTouch = 0;
    roomIn = 0;
    size = 0;
    for i = 1 to 27 do (--loops through all of the rooms
        x = DNA[d][i][5];
        y = DNA[d][i][6];
        z = DNA[d][i][7];
        width = DNA[d][i][2];
        height = DNA[d][i][3];
        length = DNA[d][i][4];
        number = DNA[d][i][13];

        alone = isAlone x y z width length height d;
        inSide = isIn x y width length;
        size += rateSize width length number;

        if (alone == false) then (
            roomTouch += 1;
        )
        if (inSide == false) then (
            roomIn += 1;
        )
```

```
    )
    extra = 0;
    for i = 28 to 42 do (
        if(DNA[d][i][5] == 0) then (
            extra += 1;
        )
    )
    --print (size as string + " " + roomIn as string  + " " + roomTouch as string  + " " + extra as string );
    return (roomIn * roomIn * roomIn* roomIn * 1.5 + roomTouch * roomTouch * roomTouch + size + 1 + extra * extra);
    --how many rooms are over
    --how may rooms are touching
    --how many rooms are too small
    --how many rooms are too big
)

function combine a b result = (
--combines DNA together to create on building.
    DNA[result] = #();

    --needs to still ask if that room created touches its parent...
    for i = 1 to 27 do (
        trial = random 1 2;
        if (trial == 1) then (
            if (isSuccess DNA[a][i][5] DNA[a][i][6] DNA[a][i][7] DNA[a][i][2] DNA[a][i][3] a DNA[a][i][10] DNA[a][i][13])
then (--ok to create
                createRoom DNA[a][i][13] result width:DNA[a][i][2] length:DNA[a][i][3] height:DNA[a][i][4] side:DNA[a][i]
[11] percent:DNA[a][i][12];
            ) else (
                createRoom DNA[a][i][13] result;
            )
        ) else (
            if (isSuccess DNA[b][i][5] DNA[b][i][6] DNA[b][i][7] DNA[b][i][2] DNA[b][i][3] a DNA[b][i][10] DNA[b][i][13])
then (--ok to create
                createRoom DNA[b][i][13] result width:DNA[b][i][2] length:DNA[b][i][3] height:DNA[b][i][4] side:DNA[b][i]
[11] percent:DNA[a][i][12];
            ) else (
                createRoom DNA[b][i][13] result;
            )
        )
    )
    for i = 28 to 42 do (
        createRoom DNA[a][i][13] result;
    )
)

function darwin_select arrayOfRatings = (
--selects from an array of values where a smaller value is more likely to be picked
--returns an array of the two items picked.
    total = 0;
    runningTotal = 0;
    m = 3000000;

    for i=1 to arrayOfRatings.count do (
        if (arrayOfRatings[i] < m) then (
            m = arrayOfRatings[i];
        )
    )

    for i=1 to arrayOfRatings.count do (
    --as bigger numbers are worse, gives inverse number making large number best
        arrayOfRatings[i] = 1/(arrayOfRatings[i] as Float - (m * 0.5));
        --adds arrat up
```


design does not improve with time

```
        total += arrayOfRatings[i];
    )

    selectionA = random 0.0 total
    selectionB = random 0.0 total
    selected = #(false, false);

    --all the ratings have been added up into a strip. The selection picks a point on that strip. A better
    --design will have a bigger area and thus more chance to be choosen. This loop cycles through the strip
    --and decides where that point is.
    for i=1 to arrayOfRatings.count do (
        runningTotal += arrayOfRatings[i];
        if ((selectionA < runningTotal) and (selected[1] == false)) then (
            selected[1] = i;
        )
        if ((selectionB < runningTotal) and (selected[2] == false)) then (
            selected[2] = i;
        )
    )
    return selected;
)

--#############################################################
--create DNA
--#############################################################
function createRandom create_txt render_CB save_CB = (
    create_txt = create_txt as Integer
    evaluation = #()
    for d = 1 to create_txt do (
        DNA[d] = #();
        for i = 1 to ROOM.count do (--cycles through all the rooms and creates them
            createRoom i d;
        )
    )
    print DNA;
    --print “sel 1”
    --print sel[1]
    --print “sel 2”
    --print sel[2]
    if render_CB == true then (
        for d = 1 to create_txt do (
            createFromDNA d (d*50);
        )
    )
    if save_CB == true then (
        saveDNA 1 create_txt false;
        for d = 1 to create_txt do (
            createFromDNA d (d*50);
        )
    )
)

function renderFromFile = (
    for i = 1 to 3 do (
        openDNA i;
        a = 50000;
        b = 50000;
        c = 50000;
        dada = 0;
        print (“###NUMBER###: “ + (i as string));
        for d = 1 to DNA.count do (
            --createFromDNA d (d*50) (i*50);
```

60



```
            a = evaluate d;
            if (a < b) then (
                b = a;
                c = d;
            )
            dada += a;
            print a;
        )
        print ("Total: " + (dada as string))
        print ("Min: " + (b as string) + " on: " + (c as string));
    )
)

function createFromFile = (
    openDNA 3;
    for a = 1 to 3 do (
        evaluation = #()
        for d = 1 to DNA.count do (
            evaluation[d] = evaluate d;
        )
        pre = DNA.count
        for d = 1 to pre do (
            sel = darwin_select evaluation;
            combine sel[1] sel[2] (pre + d);
        )
        for d = 1 to pre do ( --strips out the old DNA
            deleteItem DNA 1;
        )
        saveDNA 1 DNA.count a;
    )
)

function startInterface = (
    rollout test_buttons "Testing Buttons" (
        dropdownlist choose_dd "" items:#("Create New", "Create From File", "Render From File");
        button theButton "Start"
        on theButton pressed do (
            if (choose_dd.selection == 1) then (--create new.
                rollout test_buttons "Testing Buttons" (
                    edittext create_txt "Number to create";
                    checkbox render_CB "Render";
                    checkbox save_CB "Save";
                    button theButton "Start";
                    on theButton pressed do (
                        createRandom create_txt.text render_CB.checked save_CB.checked;
                    )
                )
                createDialog test_buttons 150 125
            )else if (choose_dd.selection == 2) then (--create from file
                createFromFile ();
            )else if (choose_dd.selection == 3) then (--render
                renderFromFile ();
            )

        )
    )
    createDialog test_buttons 150 75
)

startData ()
startInterface ();
```

Autonomous scripts

The script designed by Daniel Davis was created to determine a specific goal. Design is often discussed as a multi- faceted process. As such, design tools have often been developed to address isolated issues in design (e.g. analysis, generation, fabrication, evaluation). All of these systems segment the design process into isolated parts and consign coherent design integration to the intuitive devices of human designers. While this is often desirable, it is also worthwhile to explore completely **autonomous design systems** which perceive and act independently of human intervention. In this class of systems, the intuitive gaps must be explicitly filled and the integration of related issues must be explicitly resolved.

As we will see in the project 'Urban Agency' of Roland Snook (Chapter Practice), there are designers who make use of similar autonomous design systems. First we have to highlight a few examples and working mechanisms of autonomous design systems that are important to understand where the projects in the chapter Practice are based on.

Microworlds

For example, '**microworlds**' offer constrained environments whose very nature helps users to develop an understanding of complex causal relationships. The goal of microworlds is, simply stated, "to develop new **external systems** of representation that foster more effective learning and problem solving." (Goldin 1991) Theories of Piagetian learning have been one of the main influences in the development of microworlds. Piaget's theory of constructivism asserts that people construct knowledge about the world through experience. Microworlds can help architects learn about the **implicit logic** of organizational and material systems in buildings by constructing the logics of those systems explicitly.

Sim City & The Sims, Microwerelden

"The goal of microworlds is, simply stated, to develop new external systems of representation that foster more effective learning and problem solving."

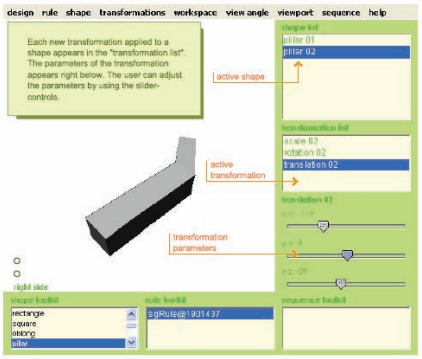
Logo (Papert), the Sim Series (SimCity, SimLife, The Sims), and StarLogo (Resnick) are computer programs that help people make sense of the world by making things in the computer. Logo was developed at MIT by Seymour Papert, a well known mathematician and advocate for computers in education. Logo is **a virtual environment** intended to teach children mathematics by allowing them to draw pictures through numerical instructions to a virtual drawing agent, the turtle. Children playing within the world of Logo learn to understand the nature of geometrical figures. (ex. a square has four equal sides and four equal angles) StarLogo is an elementary programming language designed by Mitchell Resnick for people with no programming experience. "With StarLogo, people can write **rules** for thousands of graphic creatures on the computer screen, then observe the group-level behaviours that emerge from the interactions. "(Resnick 1994) At this stage I would recommend to use the application Processor because it is more related with architectural CAD-programs.

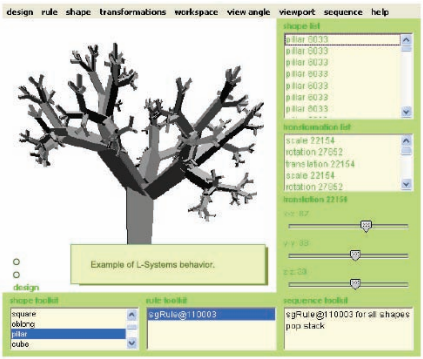
The script by Daniel Davis is based on only a few rules. In games such as 'Sim City' there are many rules written which determine the game play. So the game is ultimately based on a set of rules. As such the term '**rule-based design**' characterizes a process described as a succession of decision-steps. In a greater context these processes originate from the theory of the 'Formal Languages'. A language according to this theory is defined as an arbitrary set of words, where words are finite sequences of letters and symbols, and sets of these symbols constitute alphabets. Following are some examples of autonomous scripts based on and classified under this term 'Formal Languages'.

Shape Grammers

It is fair to say that '**Shape Grammars**' departed from this theory based on shape manipulations rather than symbols (Gips, 1975; Stiny 1975). A rule, in this context, describes a relationship between shapes. The relationship is further defined as a two-state condition: an initial configuration of space and a final one. The concept of rule-based derivation is quite simple: "under these conditions, take the following action" or "if this spatial configuration is found, then change, modify or replace it with this one". A rule therefore, describes a conditional execution of **a design action**. Ultimately, a design process is expressed globally as a sequence of rule application.

"A formal language is defined as an arbitrary set of words; words are finite sequences of letters, symbols; sets of these symbols constitute alphabets. Finally, a generative grammar produces a language (Revesz, 1983). Formal languages have been used in many fields

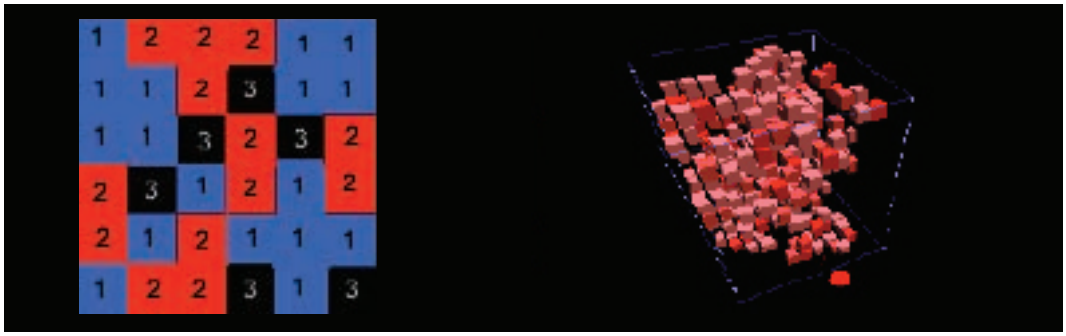




Shape Grammers, Stylianos Dritsas

Cellular Automata

Developed by John von Neumann(1903-1957). "Cellular automata are discrete dynamic systems whose behaviour is completely specified in terms of a local relation. A cellular automaton can be thought of as a stylized universe. Space is represented by a uniform grid, with each cell containing a few bits of data; time advances in discrete steps and the laws of the "universe" are expressed in, say, a small lookup table, through which at each step each cell computes its new state from that of its close neighbours. Thus, the system's laws are local and uniform." (Brunel University Artificial Intelligence Site)



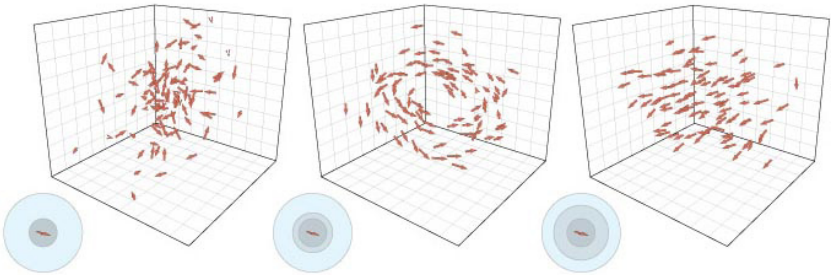
Design Tooling; S. Dritsas, A. Kilian & Y. Loukissas

Swarm Intelligence

Swarm intelligence (SI) is artificial intelligence based on the collective behaviour of decentralized, self-organized systems. The expression was introduced by Gerardo Beni and Jing

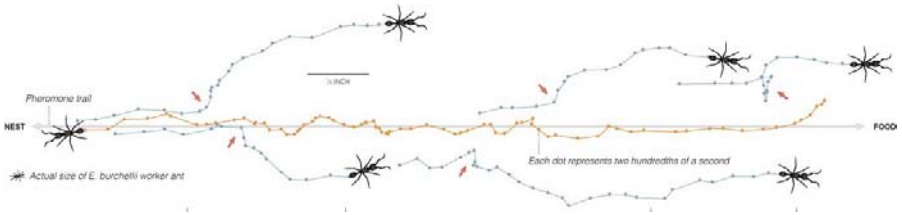
"A positive feedback mechanism ensures that, over time, a population of agents stabilise around the global-best solution.

Wang in 1989, in the context of cellular robotic systems[1]. SI systems are typically made up of a population of simple agents interacting locally with one another and with their environment. Although there is no centralized control structure dictating how individual agents should behave, local interactions between such agents lead to the emergence of global behaviour. The basic aspects are self-organization by direct communication as well as indirect communication through the environment (stigmergy). Natural examples of SI include ant colonies, bird flocking, animal herding, bacterial growth, and fish schooling.

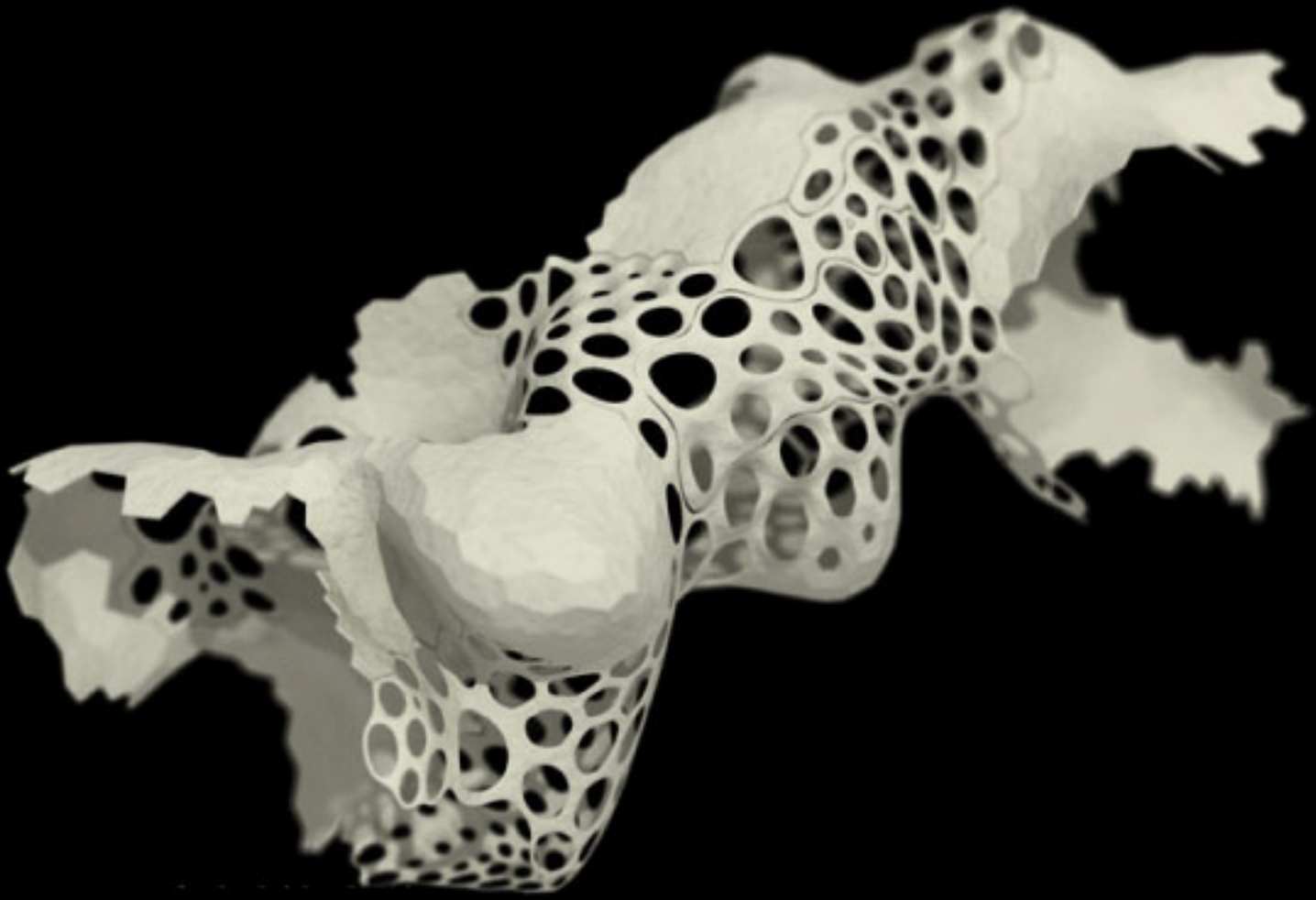


Swarm Intelligence, D. Couzin, Journal of Theoretical Biology

Ant colony optimization (ACO), Particle swarm optimization (PSO) and Stochastic Diffusion Search (SDS) are some autonomous mechanisms based on Swarm Intelligence.



ACO, D. Couzin, Journal of Theoretical Biology



67

PRACTICE

APPROACHES

Application

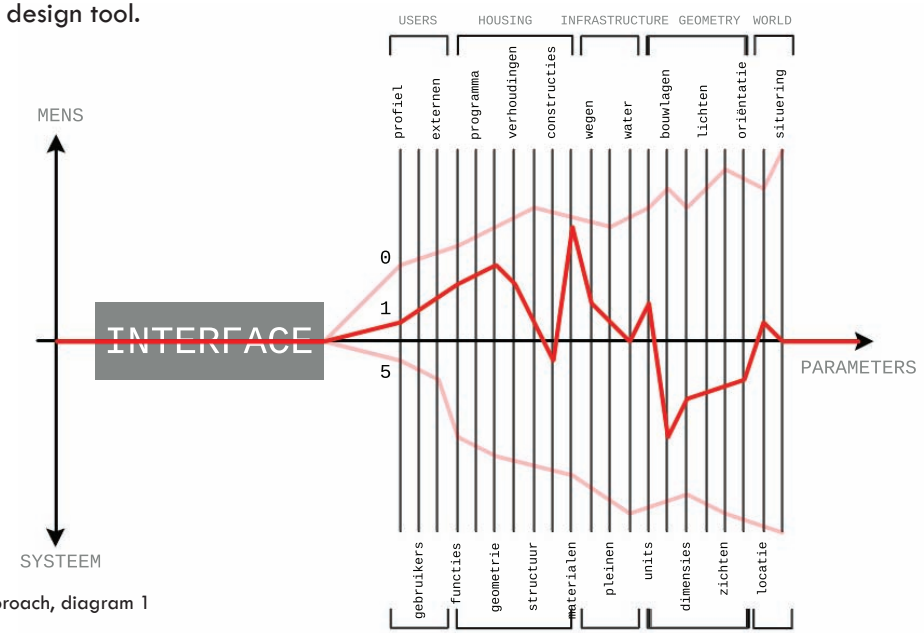
What method, what system, does an architect use to design a building? How are programmatic needs and context, with their degrees of freedom and constraints, translated into architectural design? Regardless of their complexity, the tasks and decisions involved can be formalized as an algorithm. As such, algorithms provide a framework for articulating and defining both input data and procedures. This formalization can promote structure and coherency, while systemically maintaining full traceability of all input data. In recent years, algorithms in architecture have been able to transcend their role as frameworks of formalization and abstraction. This has been made possible in a large part by the integration of scripting languages into CAD programs. Algorithms' output can now be directly visualized, enabling their use as a generative **design tool**. Since algorithms provide the benefits of scalability and permutability, multiple variations of a scheme are easily generated. A slight tweaking of inputs or process leads to an instant adaptation of output. The question arises to what extent the codification of a process through an algorithm has the ability to influence and alter the process itself. Can the structure, grammar, and logic of the language used to depict the algorithm have a relevance per se to the design, and can elements of this logic be embedded into the architecture? Can the language itself provide a basis for architecture? We approach several projects from different points of views to get a better insight on these **evolutionary design strategies**.

5 approaches

Through previous chapters it has become obvious that design is a multi- faceted process. As such, design tools have often been developed to address isolated issues in design. Michael Hansmeyer has been studying algorithms in architecture and makes notice of the following five approaches: permutation, optimization, simulation, transformation and generation. In his first approach are parameters, which are controlled throughout an interface, the most important characteristic. As such, I have named this first category as **a parametric approach**. His second approach, named by Michael Hansmeyer as optimization, is pointed out for evaluating a design through a number of iterations runned by the script, **a feedback approach** between script and designer. His third approach is called simulation. Micro worlds, already discussed in the previous chapter, is an example of simulation. Thus we can conclude that this is **an approach by means of `swarm intelligence**. Another approach is the one of

"In recent years, algorithms in architecture have been able to transcend their role as frameworks of formalization and abstraction.

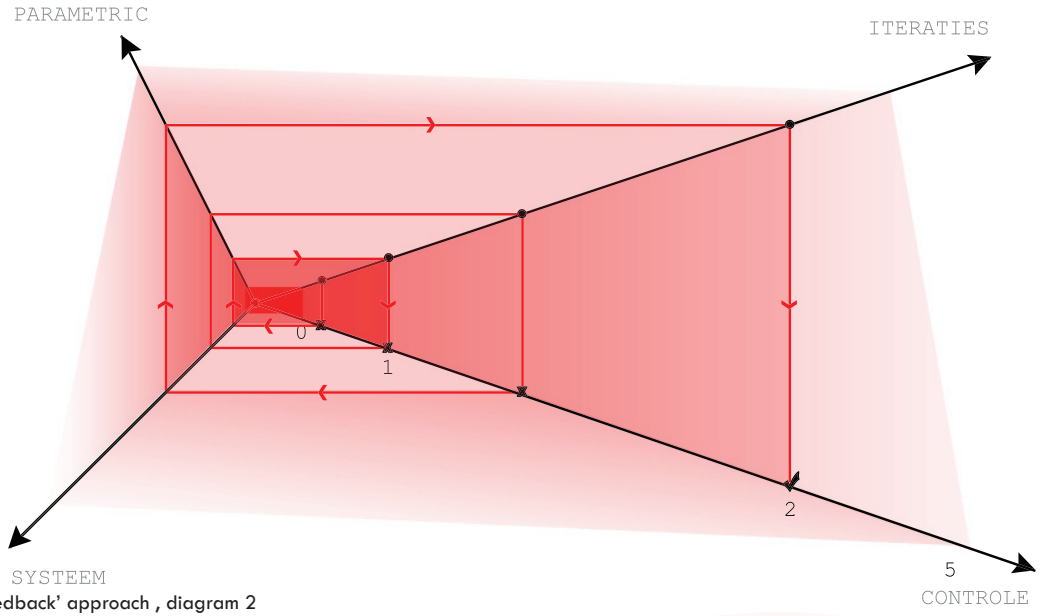
transformation, already mentioned in the thesis. Transformation is transition of information. This is what the **approach by means of `datescapes'** is all about. And the final approach, formulated by Michael Hansmeyer as generation, is a point on which architecture creates its own DNA (script) and has the possibility and capacity to grow, to develop and even reproduce itself. At present this still happens in the virtual world, but in a distant future architecture will develop techniques and intellectual capacities for itself to behave throughout its built environments. The following diagrams are my personal interpretation on these five approaches of Michael Hansmeyer and shows the relationship between the human being (i.e. the architect) and the artificial system (i.e. the computer). It indicates that we are at the beginning of an evolution within the world of design where computers are becoming more than just a design tool.



Parametric approach, diagram 1

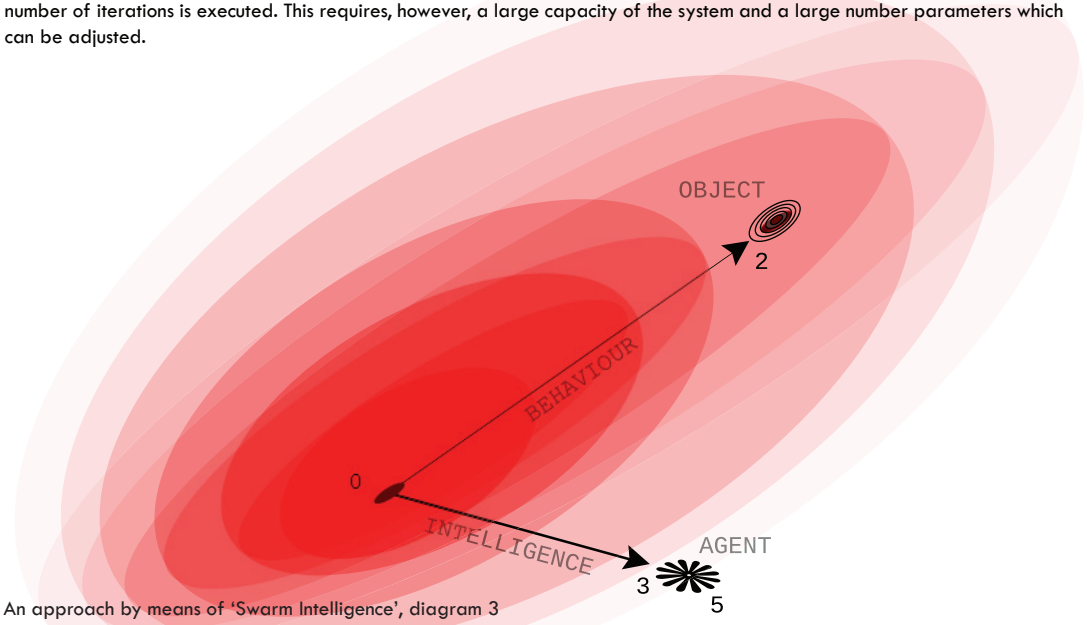
Project 1 of Evanderfeesten is situated in the centre of the diagram with an very irregular line. This means that both the man as the system (the computer) have influence and control over the parameters. After analysing project 5 of R&Sie(n) and placing it in the same diagram we notice that in the system now takes control of almost all of the parameters. If we draw a line of a project which is approach by the conventional way of designing (0) in this diagram we notice the opposite.

Michael Hansmeyer has been studying algorithms in architecture and makes notice of the following five approaches: permutation, optimization, simulation, transformation and generation.



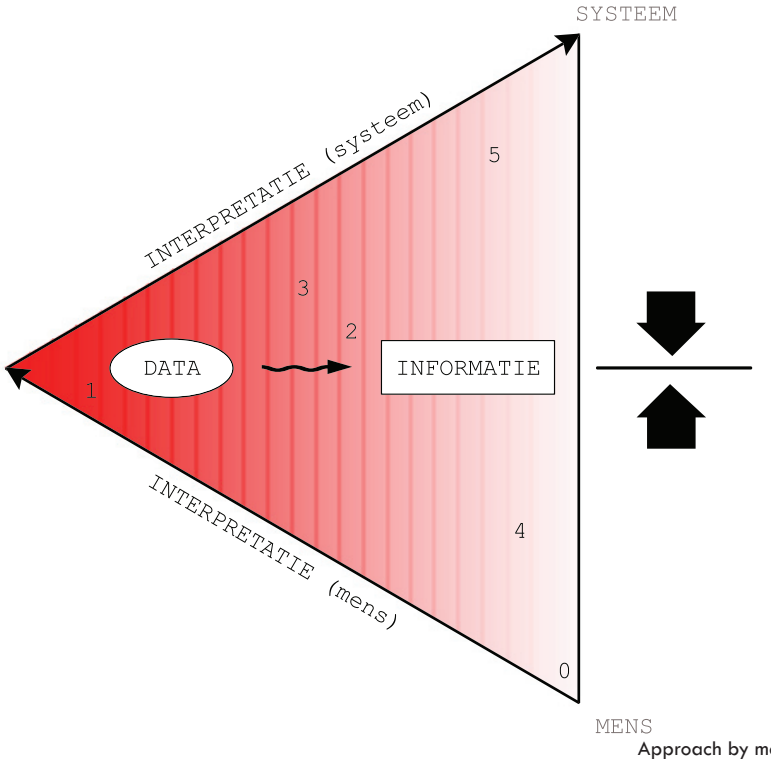
A 'feedback' approach , diagram 2

Each project in this diagram starts from the origin. If we want to optimise a project according to certain fitness criteria we will have to execute several iterations. If we do this in a conventional way (0), we are limited to a small number of iterations. If we use the computer as a partner (for example in project 2 and 5), we notice that because of it's calculation ability an infinite number of iterations is executed. This requires, however, a large capacity of the system and a large number parameters which can be adjusted.



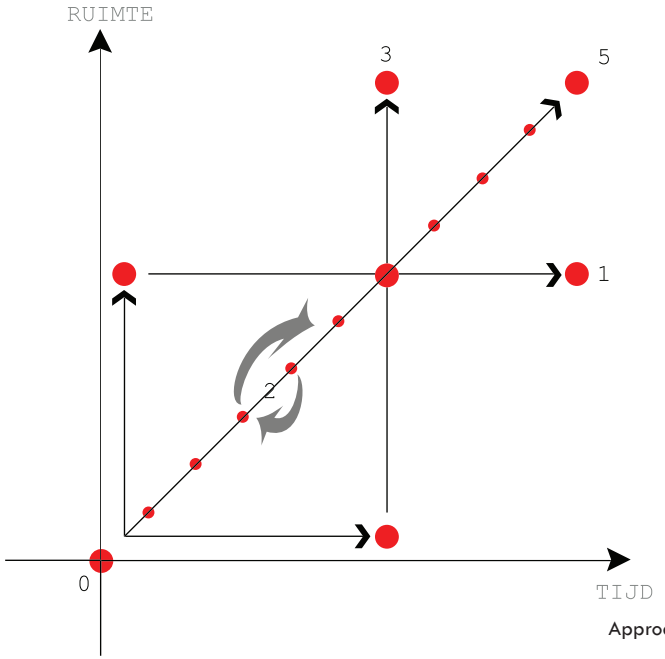
An approach by means of 'Swarm Intelligence', diagram 3

In project 3 of Kokkugia 'agents' determine the design. Other designs are based upon less intelligent 'objects'. These objects contain only a small amount of rules and they have no ability to communicate or move within the 'hybrid space' (2). In reality (0), we find similar methods like sensors in a building to control the ecosystem.



Recent decades we have continuously tried to store information on computers. Computers stores this info as numeric data. And today we want the computer to interpret this data. We notice that in project 1, 2 and 3 with only data is used as a reference. Although it is necessary, or even required for a project to base it on relevant information (4). Just as in diagram 1, it is necessary to look for a balanced status.

Approach by means of 'datascapes', diagram 4



In project 1, we notice that once the computer has executed the script the form of the design is determined. However, it is possible that each generation of the same script generates a different form. In 3 project we notice that the script, or the 'agents', has moved in time until a certain condition is found. In this 'hybrid space' architecture is dynamic. As in previous diagrams it is once again needed to search for an balanced status.

Approach with the intention to generate, diagram 5

Interpretation

One must well know that each project, discussed in this part, can be approached by each of these categories. To prove this I have made a visualization of each approach in which every project can position itself. A conclusion follows: we see that every project uses all approaches in several ways, what of course depends on the original objective of each project. Every project is discussed in five steps: the assignment (what is the objective?), the approach (what vision?), criteria (how the process works?), conclusion (summary) and a reflection (personal critics).

Project 1:

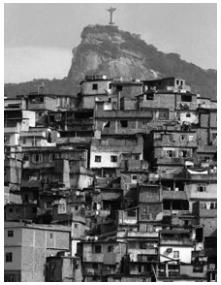
'Confection for the masses in a parametric design of a modulor favela structure', Evanderfeesten

"Parametric approach"

I have started to analyse this project because it departs from the `favela' structures whereas I was attracted to because during six months of my studies and thesis research I have been living in **South-Africa** (Port-Elizabeth) where similar structures ('shaqs') form images covered over a beautiful landscapes where life is lead by poverty and illness.

Assignment

The objective of this project is inspired on **the favela structures** in Sao Paulo, Brazil. The pattern of these areas shows an uncontrolled growth which leads to a vast, seemingly chaotic, but in reality very structured manner of living. The analysis of this pattern of structures has inspired Evanderfeesten to develop **a modular system** of building blocks. The uniqueness and identification is not only present in its cosmetic skin, but also in the arrangement and combination of the building blocks.



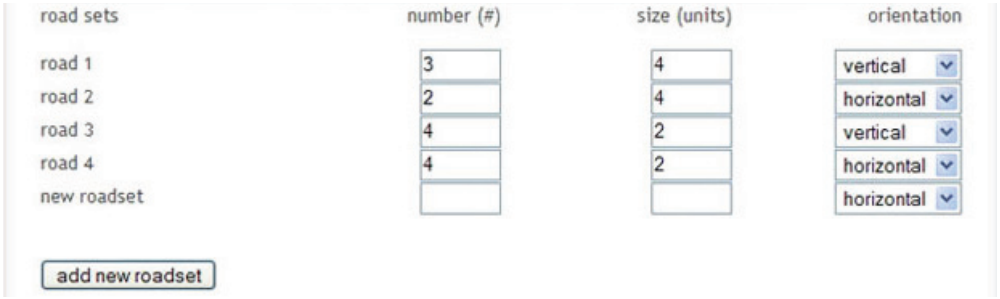
'Favela' structures, Sao Paulo, Brazilië

"The uniqueness and identification is not only present in its cosmetic skin, but also in the arrangement and combination of the building blocks."

Location, Eindhoven, Nederland

Approach

The applied method for creating this new environment is based on parametric design. Within the parametric procedure certain **parameters** are given before the script is executed. . The essential elements are mostly derived from either the building blocks or the urban context. The manuscript interprets the input parameters to produce a model that fulfils all requirements. The produced results serve as **a starting point** for further design.



Set-up Interface, Road Parameters

Criteria

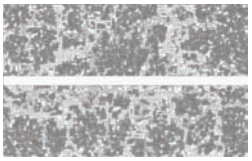
The script starts with **a set-up,** first a limited number of parameters is established out of a PHP based web page. Afterwards, these are stored in MySQL database. During the calculation stage the parameters are picked up from that PHP based database and calculations are being made. The results is again stored in a MySQL based **database**. During the drawing stage all geometrical information is picked up and made ready through a conversion script for a CAD-programme, in this case: Rhinoceros. After all **geometrical information** is drawn the model is stored in a file and is rendered in 3d Studio Max.

"The applied method for creating this new environment is based on parametric design. The essential elements are mostly derived from either the building blocks or the urban context.

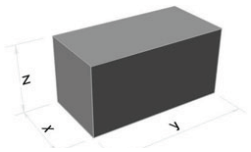
Site

Spaces

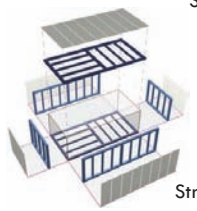
Infrastructure



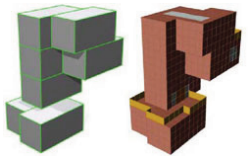
Modules



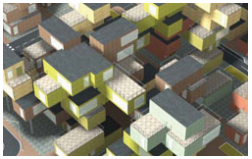
Module



Structure



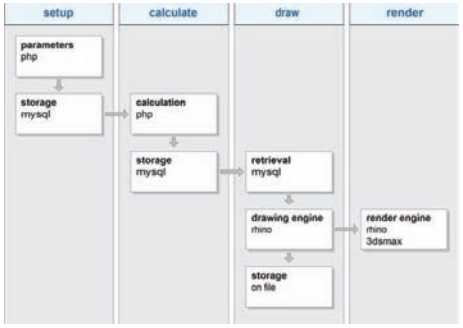
Construction



System

Conclusion

The manuscript treats a wide range of structural elements within a grid. The activities concerning the infrastructure, the green areas and leisure places, the commercial areas, the future development areas and the modules for housing are all elements witch use the same algorithms but are produced with different input parameters.



Scripts

Modular system

Reflection

The methods used could change the role of the architect in the design the design process . The produced design that reflects environments of mass housing requires new standards for social interaction and structuring building blocks in a high populated urban context. Thus it does not satisfy to the high standards in which we are use to live today, know as the Western world.

The produced design that reflects environments of mass housing requires new standards for social interaction and structuring building blocks in a high populated urban context.

'Algorithms in Architecture: optimization',

Michael Hansmeyer

Project 2:

Approach by means of "feedback"

Assignment

Design a commercial **apartment building** for lower Manhattan that contains units of various sizes.

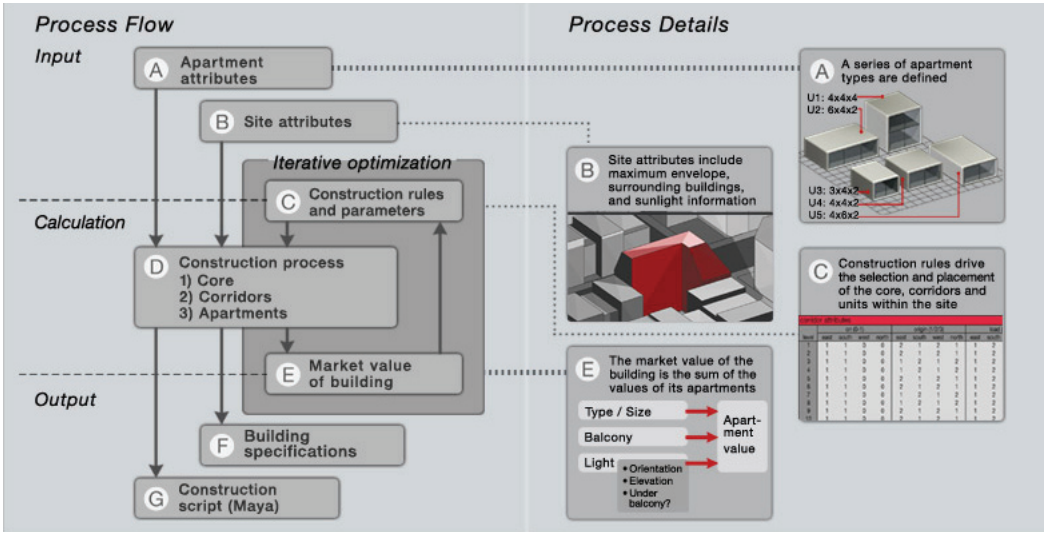
Approach

What is the role of an architect in today's New York housing market? With few exceptions, current apartment buildings are identical concrete skeletons with only minor variations in their curtain wall facades. It is mostly in their interior portioning that they differ; the internal spaces – largely disconnected from both structure and façade – are configured last minute to best address the latest **market demand**. From this thought the project is concentrated pure on partitioning units. The building's design is reduced to an optimization problem in which the site's maximum envelope is segmented and filled with the optimal combination and configuration of apartments. The **optimum** is regarded as the highest possible market value for the building that this segmentation can achieve.

Criteria

The algorithm has **three fixed inputs**: A specific site and its attributes, a catalogue of apartment types of different sizes, and the apartments' price sensitivies to various factors. The process' variable input are the actual construction rules that determine the placement of the building's components. Parameter ranges for these rules are defined. These include, for instance, the possible lengths of corridors, the number of corridors that can emanate from the core at each level, and wether corridors are single or double loaded. The calculations consist of two steps. First a building is **constructed** based on the construction rules and the values chosen from within their parameters ranges. Second, the variant is **evaluated** by calculating the combined market value of its apartments. A genetic algorithm plug-in changes the values of the construction parameters after each **iteration** in an attempt to find better solution than the previous variant. This process is repeated thousands of times until no better combination of construction parameters to increase market value can be found. At this point, the algorithm produces a script to visualize the optimal variant in a CAD program. It also produces building specifications that can form an input for further algorithms.

"With few exceptions, current apartment buildings are identical concrete skeletons with only minor variations in their curtain wall facades.



'Feedback' process, Michael Hansmeyer

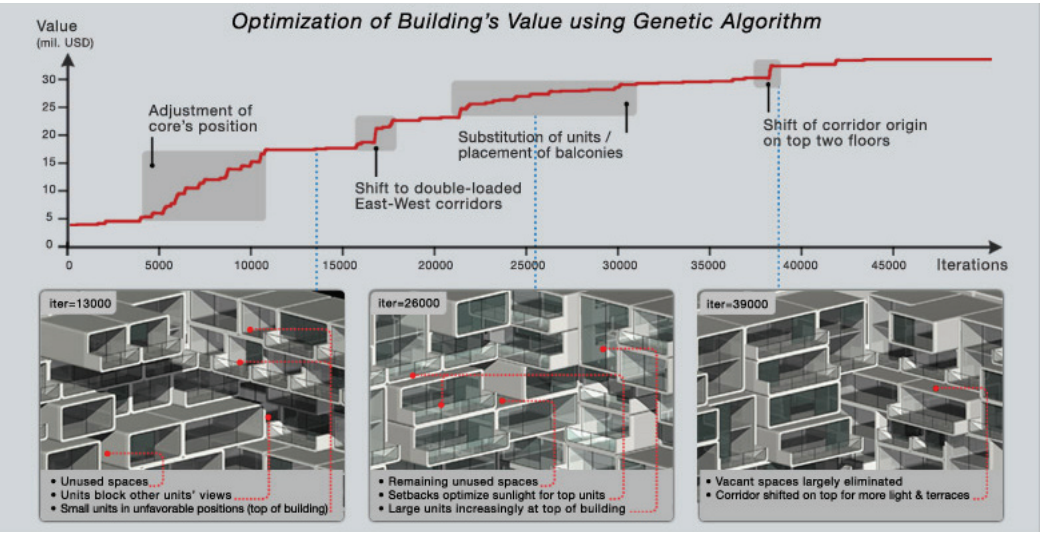
Conclusion

The algorithm in this project both generates and evaluates a building and does so recursively. The process needed approximately 40.000 iterations until a near-optimum was reached. The algorithm produces not only the market value of the building which acts as a reference point for further iterations, but it also provides detailed building specifications that can constitute inputs for further secondary algorithms such as calculation of a structural system. While the optimization leads to generation of building's form, this shape is limited by the construction rules and their parameter ranges, which in this case prescribe an assembly of pre-defined units.

Reflection

Designing a building based on the highest market value is an approach of a broker. Orientation and dimensions are just some of the important factors which determines the qualities of an apartment building. Views from within a unit to the outside space are also determinative qualities. Private and public areas between several units are even more important. An alternative unit which is generated from the optimized process, must therefore be still evaluated based on other requirements besides the market

"The building's design is reduced to an optimization problem in which the site's maximum envelope is segmented and filled with the optimal combination and configuration of apartments.



Optimalization process, Michael Hansmeyer

value and the dimensions. The algorithm is generating tool for some market factors during a design process, thus it can be a reference starting point if the assignment requires certain defined constrains.

'Urban Agency', Feidad 2005

Project 3:

Kokkugia, Roland Snooks

Approach through the use of "Swarm Intelligence"

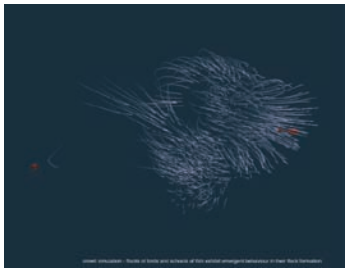
Assignment

The **proposition** of the project is that an organism capable of autonomous or intelligent drift through the urban fabric is able to develop **an emergent architectural form** and organisation intrinsic or peculiar to its environment. The project posits a strategy based on a far-from-equilibrium thermodynamic model, where it is the drift of information and behaviour through a population of design agents that is capable of self-organising into new structures and form.

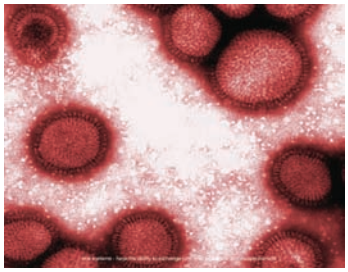
"This is not a proposal for an architecture which drifts, but rather a methodology which drifts within the space of design."

This drift can be thought of similar in operation to that of the spread of panic through a crowd or heat transmission though gas. However its motives are closer to that of the 'dérivé' technique of the derive to overcome the prescribed use of the city - similarly drifting design agency is an attempt to subvert the prescribed architectonic and programmatic relationships. By drifting the organism extracts and exchanges information with the city which feeds and influences its desires. the organism isn't simply playing out a set of behaviours but their behavioural combinations are capable of morphological development. The agents motion is controlled by these recombining desires specific to site and the intra-relationship of the swarm, however in doing so there is something of a cross pollination of architectonic and programmatic information.

The **program** or brief for the project is a networked headquarters for an organisation set up to explore biopower - comprising agencies from the UN, NATO, medical research organisations etc. This organisation is intended to operate more as an organism than a traditional building and as such the design strategy knits or weaves the organization or organism into the urban fabric. This is not a proposal for an architecture which drifts, but rather a **methodology** which drifts within the space of design.



Flock formation, School of birds



Exchanges, Viral Systems

Location Mid Town, New York City

Approach

'Agent drift': The drifting design agents get their characteristics from three types of agent based simulations: crowding and flocking, convection systems and viral systems. The design agents posses a set of attributes which determine their behaviour, as well as the architectonic and programmatic information or DNA that they deposit to generate architecture. Their behaviour and movement is determined by a scripted set of responses to these attributes. In

"In this project the city is not modeled as built form but rather as millions of packets of information which contain the attributes of the built environment.

this project the city is not modeled as built form but rather as millions of **packets of information** which contain the attributes of the built environment (programmatic and architectonic). The design agents navigate this **datascape** exchanging information with the environment in generating context responsive architecture. in this exchange the agents carry out certain agendas relating to cross programming and subverting the current relationship between public and private space.

Accretion: The design agents generate architecture through a set of emergent actions, involving accretion or depositing and mutation. The primary methodology for depositing architecture is through the generation of **voronoi cells** which become the basis for the parasitical form. A voronoi algorithm traces a plane which is equidistant from any two points and as such generates a set of cells which are in a geometric state of equilibrium regarding the locations of the design agents. The voronoi cells are then the base geometry for inserting: form, structure, program, or subtractive form (boolean difference).

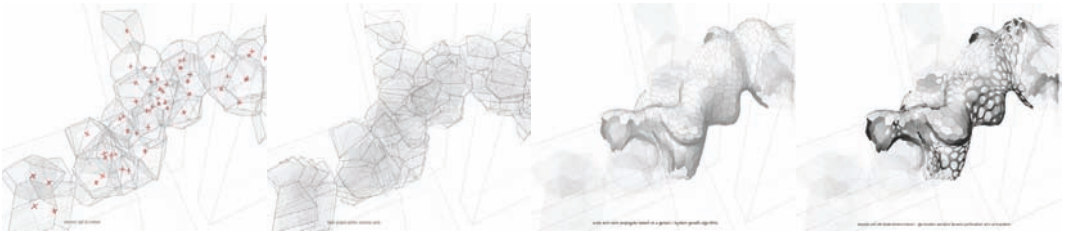
The second action undertaken to generate architecture by the design agents is that of **transformation**. This strategy, drawn from viral systems, transfers information from the agent to the packets of information or parameters controlling the architectonic and programmatic form of a building, and in doing so generates morphological change - a mutation. This technique is most evident in the transmission of architectonic information controlling facade systems, in particular it is used to exchange: facade perforation ratios, level of resolution (smoothness) and materiality as well as attributes peculiar to the agents such as turbulence.

A **algorithmic code** is employed to generate an outer facade, which grows over the deposited voronoi cells. The cells are parsed though a tomographical procedure which extracts a threshold of density surrounding the cells and generates a zone on which the cellular facade propagates. The morphogenetic process of facade propagation is generated through a lindenmayer system, whose substitution rules underlie the growth algorithm. While this system is genetically motivated, the phenotype is able to respond to its context. These cells then become the input for greater specialisation and local differentiation through the transmission of their attributes. The exchange of attributes generates a high level of differentiation within the facade, while maintaining a continuity of character.

drift simulation - design agents navigate the architectonic information of the city

Conclusion

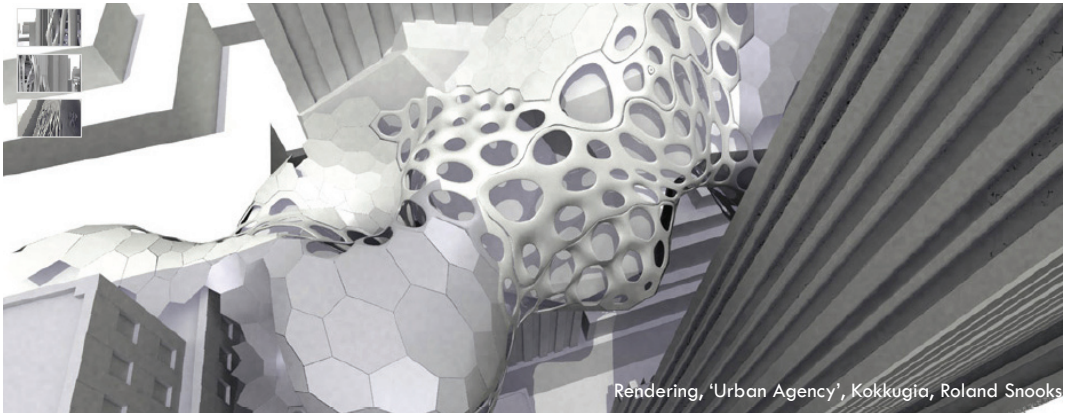
The agents have a series of **attributes** such as program and architectonic information which is exchanged between the agents and the existing built fabric in a process similar to convection. The agents **behave** differently depending on what attributes they possess - consequently there is a self-organisation of the architectonic and programmatic information. In doing this, the methodology creates a an architecture which is specific to its location.



'Agents drift' Program Voronoi structure Facade structure

Reflection

The system involves constant feedback as every agent is checking all the agents within a certain vicinity at each step. This project was done in Maya using a plug-in called 'ai.implant' which has a scripting language very similar to Mel. However since completing this project Kokkugia has done a lot of work with agent modelling using Processing and Roland Snooks recommended me that if you are going to do agent based work that you should do it in a very lightweight environments like processing rather than in something like Maya. However for projects which are more geometry intensive, he recommends learning Rhinoscript.



Rendering, 'Urban Agency', Kokkugia, Roland Snooks

"By drifting the organism extracts and exchanges information with the city which feeds and influences its desires.

'IFCCA', Competition New York 1999

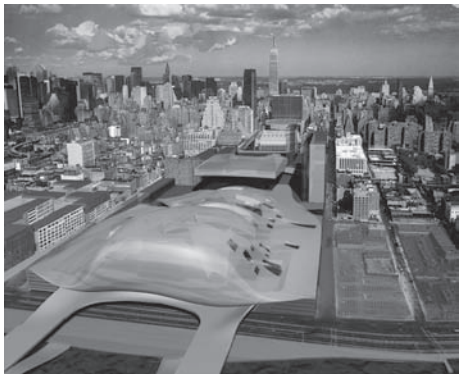
Project 4:

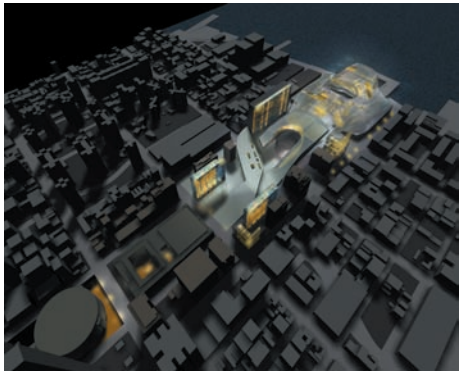
UN_Studio, Ben van Berkel

Approach by means of "datascapes"

Assignment

This proposal studies the Competition Site in relation to its historical development, to the greater metropolitan area of New York City and especially to its immediate surroundings. The subject of this study is the area between 42nd and 23rd Street. This area contains a number of service facilities that form obstacles in two respects: they constitute **physical barriers** that block fluent connections between the locations and they prevent the further development and full land use of several districts. The site has been affected by the decline of the harbour since the 1960s.



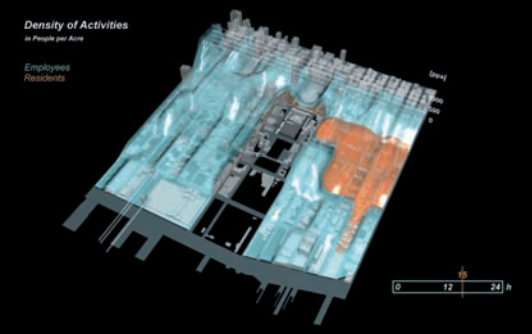


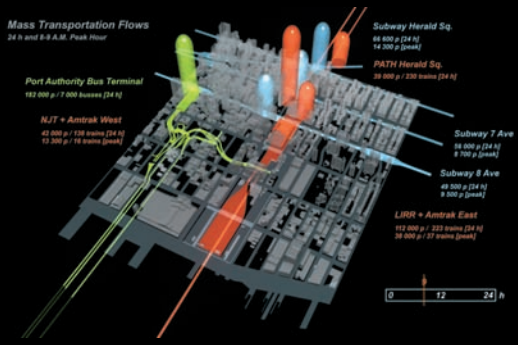
Renderings

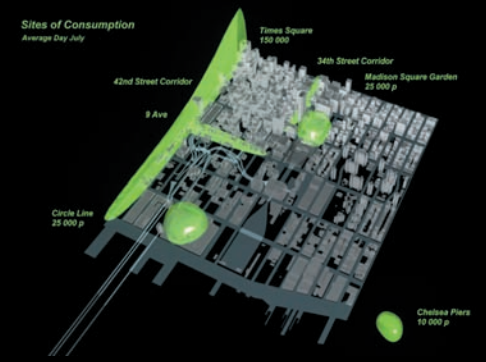
Approach

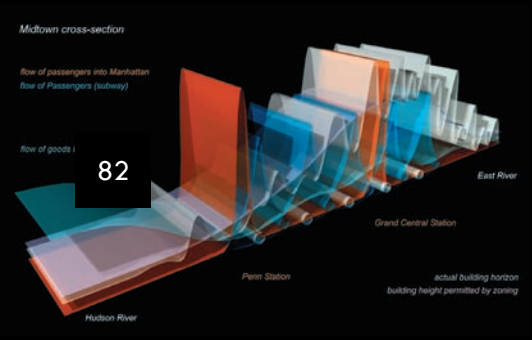
The answer is to develop and implement a new **urban package** for the postindustrial, global city. The site is acting as an infrastructural link within a broader network of sites of attraction, sites of consumption, major transportation nodes. It has the potential to function as a lobby for Manhattan. The consideration at the basis of this scheme concerns the future of this part of Manhattan in the face of the increasing homogenization of cities. All over the world similar metropolitan conglomerations cater to a **transnational population** of urban travellers. These **non-places** of the business elite all seem based on the original prototype of Manhattan, inviting the question what step Manhattan will take to distinguish itself from its replicas and to once again set the tone for the future.

"The critical package thus amounts to the articulation of an urban policy, consisting of a critically generated engagement with the situation in which the production of the city takes place.

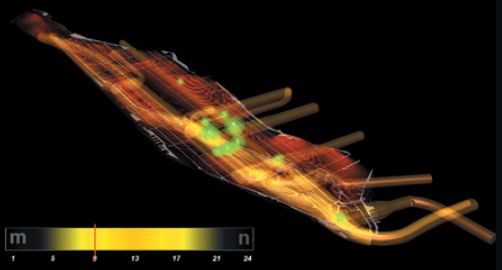

Density of Activities
in People per Acre
Employees
Residents
0
12
24
h

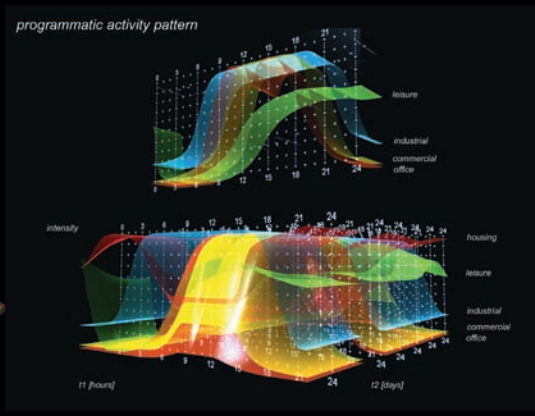

Mass Transportation Flows
24 h and 8-9 A.M. Peak Hour
Subway Herald Sq.
66 600 p [24 h]
14 100 p [peak]
PATH Herald Sq.
39 000 p / 230 trains [24 h]
Port Authority Bus Terminal
182 000 p / 7 000 busses [24 h]
NJT + Amtrak West
43 000 p / 138 trains [24 h]
13 300 p / 16 trains [peak]
Subway 7 Ave
56 000 p [24 h]
8 700 p [peak]
Subway 8 Ave
49 500 p [24 h]
9 500 p [peak]
LIRR + Amtrak East
112 000 p / 323 trains [24 h]
38 000 p / 37 trains [peak]
0
12
24
h

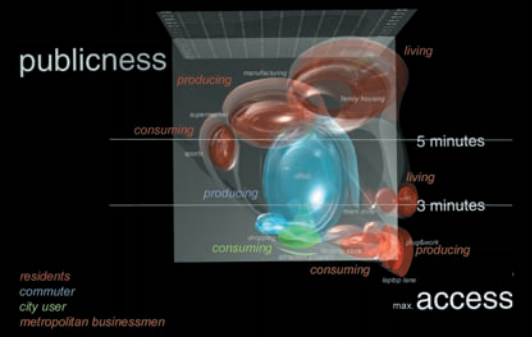

Sites of Consumption
Average Day July
Times Square
150 000
34th Street Corridor
Madison Square Garden
25 000 p
42nd Street Corridor
9 Ave
Circle Line
25 000 p
Chelsea Piers
10 000 p

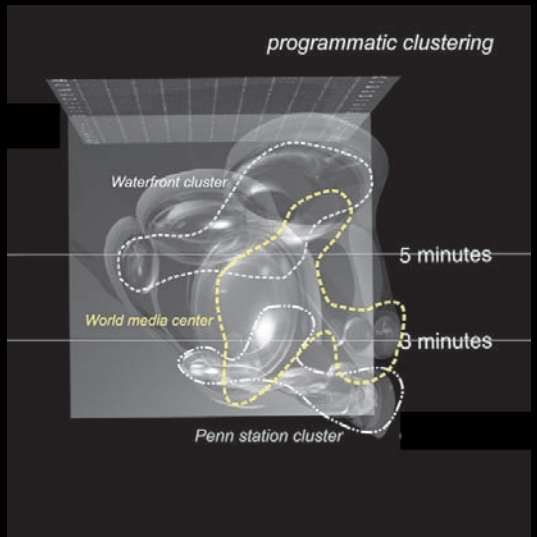

Midtown cross-section
flow of passengers into Manhattan
flow of Passengers (subway)
flow of goods i
East River
Grand Central Station
Penn Station
actual building horizon
building height permitted by zoning
Hudson River

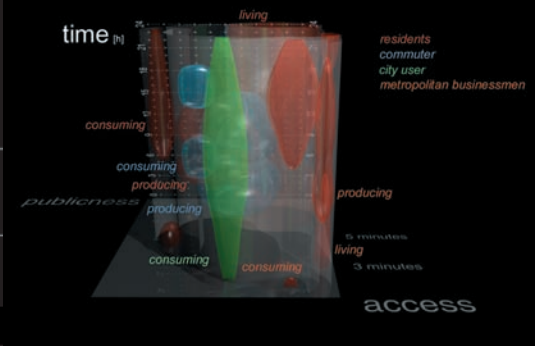
82


m
n
1
5
9
13
17
21
24


programmatic activity pattern
leisure
industrial
commercial
office
intensity
housing
leisure
industrial
commercial
office
t1 [hours]
t2 [days]


publicness
living
producing
consuming
5 minutes
living
producing
3 minutes
consuming
producing
consuming
residents
commuter
city user
metropolitan businessmen
max.
access


programmatic clustering
Waterfront cluster
5 minutes
World media center
3 minutes
Penn station cluster


time [h]
living
residents
commuter
city user
metropolitan businessmen
consuming
consuming
producing
publicness
producing
producing
5 minutes
living
3 minutes
consuming
consuming
access

Criteria

The critical package for the global city is built up using **scenarios, diagrams, parameters, formulas and themes**. Diagrams are made which map the performance of Manhattan in order to extract parameters for the development of the site. At the basis of the critical package lies the question, which is the optimal combination of factors for the site to function effectively with respect to programs, construction, economy, public and private desires, community concerns, and political and managerial feasibility. The term critical package is chosen to indicate relational qualities; any intervention has reverberations on many levels, and therefore in reality always constitutes a package of measures and decisions. Our proposal is presented as an integral, inclusive strategy, which allows fragmentation and difference to be absorbed into a coherent, continuous approach, aiming for continuous development. The **critical package** thus amounts to the articulation of an urban policy, consisting of a critically generated engagement with the situation in which the production of the city takes place.

Conclusion

The proposal is based on the concentration of **transportation** and other **service nodes**, which frees land and money for new developments, and which improves local connections by dissolving the physical barriers formed by these service and transportation areas. Although our proposal amounts to a radical intervention, it is designed to be carried out in phases and includes possibilities for temporary uses. The proposal moreover is intended as strategic and visionary and emphasizes the need for **datascapes** to uncover the shared values inherent in the various long-term, site-specific interests. Programmatically, the World Media Cluster contains the highest variety of uses.

Reflection

In this project graphical techniques has been explicitly used to extract **data** out of information which can be used for further processing in the design. These graphic techniques are visualised through draw applications but have not been by generated by it. We notice therefore that a standard computer is not able to understand information. The designer does and extracts relevant datasets which can be used for further development of algorithms. In M. Hansmeyers' projects 'Algorithms in Architecture cfr. transformation' we see that these datasets are not generated by the computer into an **informative model**. It is exactly here that an architect, with a post-modern mind, comes into play. He has to the capacity to work **metaphorically**. Will our partner, the computer, ever be able to develop such an intelligence. According to me, this will be one of the questions which remain unanswered at the end of this thesis and possibly form a basis for further research…

"It is exactly here that an architect, with a post-modern mind, comes into play. He has the capacity to think conceptual.

Project 5: *'I've heard about…, a flat, fat, growing urban experiment',*
R&Sie(n), Francois Roche

Approach with the use of "generative algorithms"

Assignment

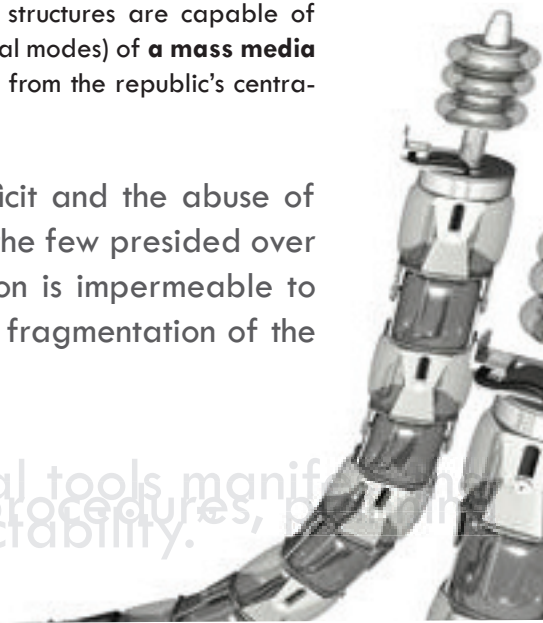
Can we envision something totally different, **urban structures** driven by human contingencies? Can we work out **adaptive scenarios** that accept unpredictably and uncertainty as operating modes? Can we write the city based on growth scripts and open algorithms porous to a number of real-time inputs (human, relational, conflictual and other data) rather than trying to design an urban future formatted by rigid planning procedures?

Approach

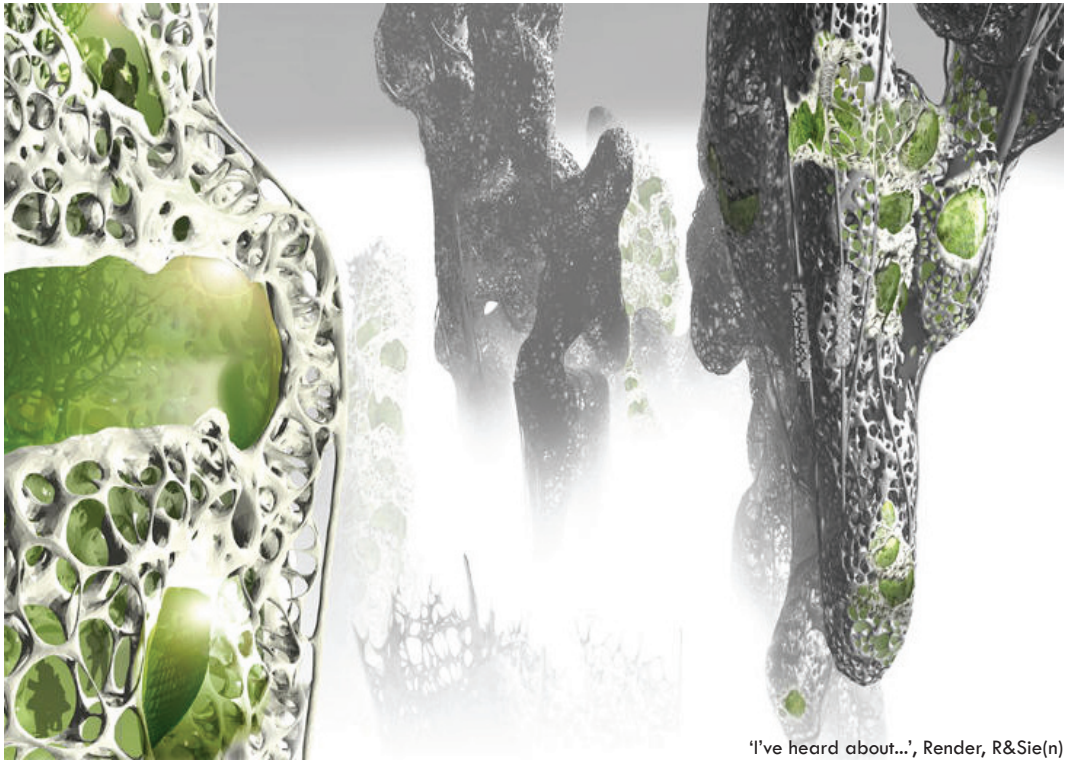
The contemporary city's developmental tools manifest the tyranny of tightly scripted **determinist procedures**, planning mechanisms based on predictability. The city's growth, densification and entropy are driven by pre-set and invariable geometrical projections. Urban morphological transformations are supposed to follow **closed scenarios** that cannot deviate from the pre-programmed representations on which they are based. Thus the cartography of the city's becoming is fettered by a mode of production that takes the future as already written. Everything yet to come is spelled out in advance and tightly locked up by that forecast.

The contemporary (European) city is formatted under Windows, unable to access the programming source codes (Linux). There no reason to believe that the "everything under control" operating modes that condition the production of urban structures are capable of reflecting the complexities (the intertwining of issues and relational modes) of **a mass media society** where the multitude of citizens is gradually taking over from the republic's centralized authorities.

"The city's making suffers from a democracy deficit and the abuse of tools that date back to a time when the reason of the few presided over the destiny of the many. The city's very constitution is impermeable to the social shifts brought about by the dilution and fragmentation of the informational and productive mechanisms."



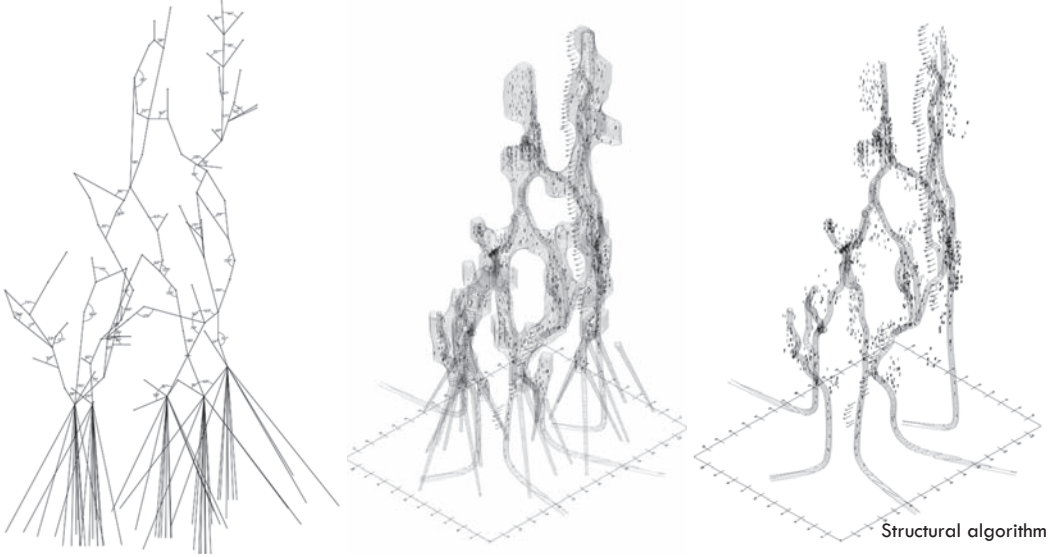
VIAB is developed in partnership with the USC/Los Angeles robotics lab. It takes its name from the terms viability and variability. The machine's purpose is to build the biostructure in real time through the secretion of the structural material that serves as the project's envelope. The VIAB is a kind of parasite, or like the polyps that live inside and are supported by the coral reef they generate. The basic idea is to design an architectural structure that is always under construction, combining incompletion and self-determination as its constituent parameters of which the VIAB is the vector. Growth, or rather the variability of growth, is to be determined by a succession of constraints arising from: structural resistance ,accessibility and measures entrained by use modes, the stress felt by the residents and swarm intelligence (collective behaviour).



'I've heard about...', Render, R&Sie(n)

"The basic idea is to design an architectural structure that is always under construction, combining incompletion and self-determination as its constituent parameters of which the VIAB is the vector.

The main **algorithm** derives from structural considerations and controls the viability of the development. Individual and collective stress are taken into account as viruses inflecting the basic construction data. As a result, the VIAB is empathetic, receptive to the subjectivity of the residents. The process of construction reveals the state of social behaviours in a constant state of negotiation. It is always a work in progress, with no attempt to predict or plan the morphological results. It is the nerve ending of **the species** that inhabit I've heard about. The VIAB is designed as a kind of phasmid, a stealth insect that looks like a stick attached to the structures that generates itself.

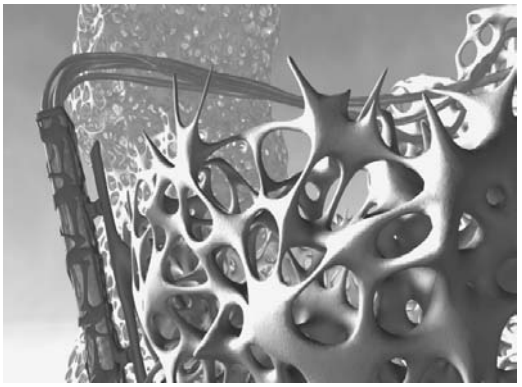


Structural algorithm

I've heard about is a fractal structure made quite literally of contingent secretions. Its architecture is based on the principles of random growth and permanent incompletion. Growth is based on negotiations between neighbours and other residents, and at the same time subjected to **collective constraints** (accessibility and structural contradictions). The urban form no longer depends on the arbitrary decisions or control over its emergence exercised by a few, but rather the ensemble of its individual contingencies. It develops by successive scenarios, without planning and without the authority of a pre-established plan. Its physical composition renders the community's political structure visible. The proliferating, coral-like

"The urban form no longer depends on the arbitrary decisions or control over its emergence exercised by a few, but rather the ensemble of its individual contingencies."

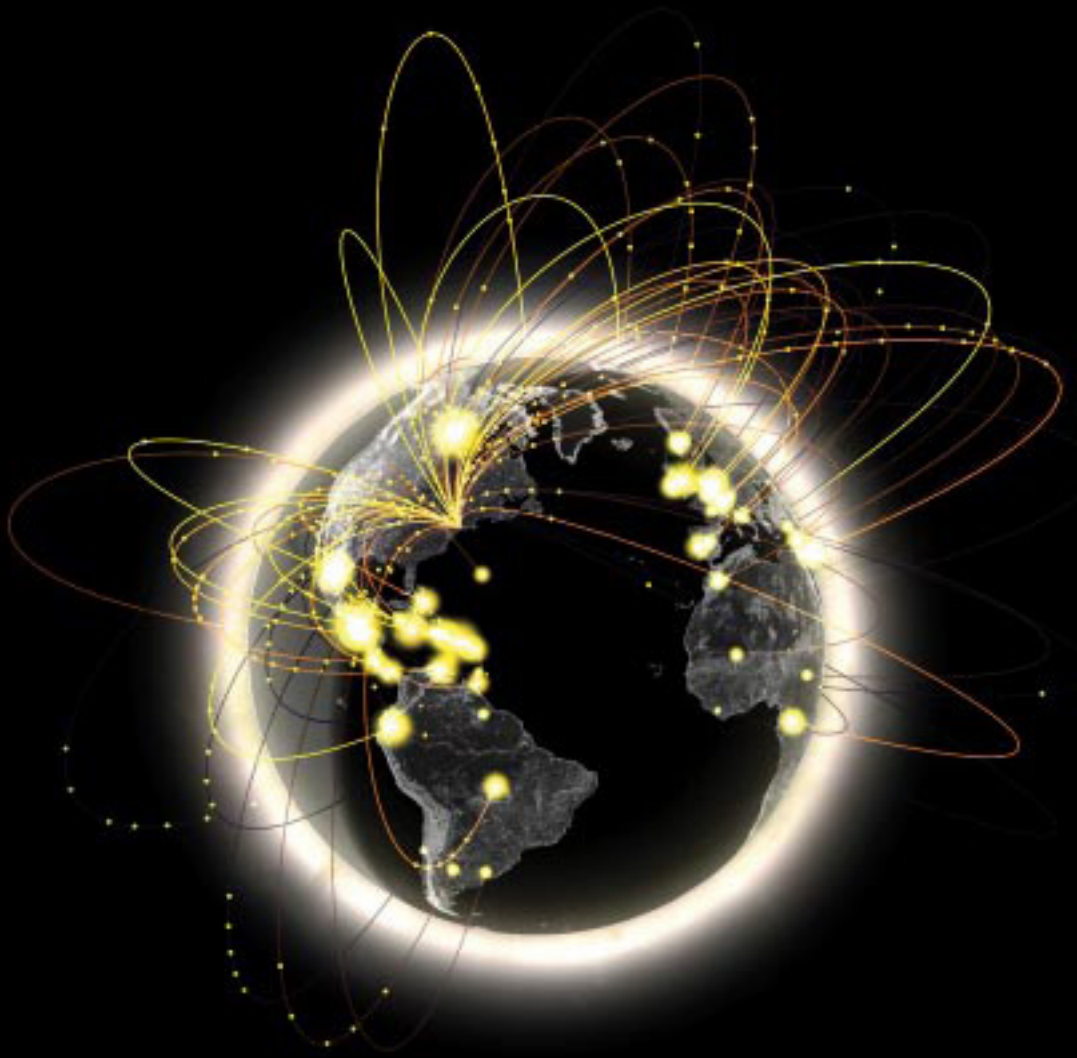
network is constituted of both imported raw materials and local materials that have been recycled, synthesized and polymerised, resources arising from the animal and vegetable species that inhabit it. Operating anthroposophically, it generates modes of exchanges, flows and blood vessels.



Construction

VIAB

I've heard about recognizes and builds on the idea of an ever-emerging, shifting and above all **fragile sociality**. Section by section, the raw materials of the habitats undergo necrosis every ten years in order to avoid an overly permanent occupation and an attendant sense of individual ownership (the early cycles will be more aleatory). A territorial **protocol** on generative and resulting diagrams had to be established. A `Protocol `is a behavioural agreement, generally in the form of a number of steps. And only if all parties behave according to the same agreements it is possible to coordinate their operations. I've heard about does not eradicate the pre-existing city but rather forms **a sedimentary deposit** over it, like Constant's New Babylon. It can be described as **a plug-in** inserted into the urban fabric.

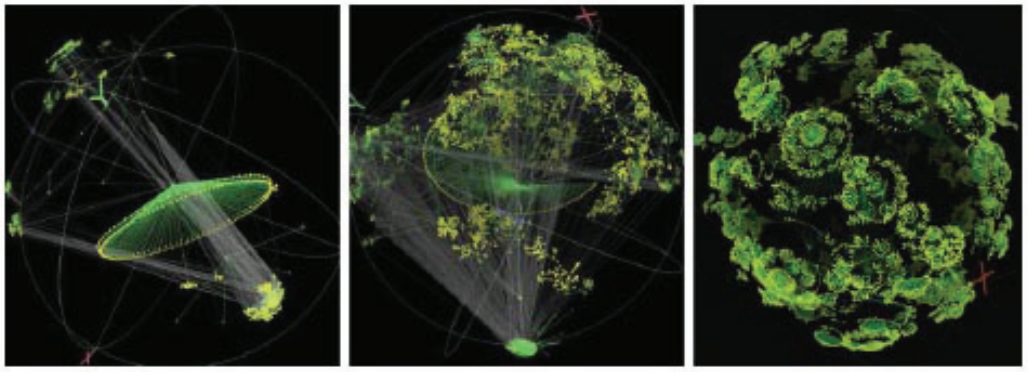


8 9 REFLECTION

CONCLUSION

Computer

Within a short space of time the computer has become a widely accepted **feature** of architecture, both in the design process and in the everyday operation of buildings and the city, and we are constantly aware that the computer's introduction into architecture will eventually have far-reaching consequences. After all, the current revolution is not just about the computer as a tool but about its role in the global telecommunications network. Since its creation architects, artists, media designers and theorists have speculated about the **ramification** of the computer. It is a theme that invites speculation, experiment and play – but that is not the only reason for the persistent questioning. Today we are **aware** that we cannot foresee all the implications of the technological revolution. We remember how long it took for people to grasp the social changes of the Industrial Revolution and how another fifty years passed before they started to think about its cultural **impact**. However skeptical or critical one might be, no one wants to be accused of having "eyes that do not see," as Le Corbusier did in his 1923 Towards A New Architecture. On the other hand – at least since Manuel De Landa's recent concept of a nonlinear history – we are aware that innovations and inventions need time to incubate, and their effects on the organization of society can be completely unexpected. The radicalization of modernity that has been triggered by the computer means that it has become increasingly difficult to fall back on traditions: more than ever, we must reflect on what the future will bring.

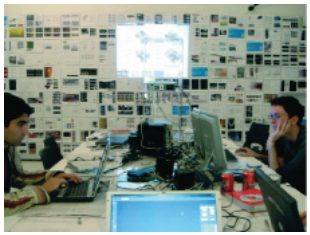


Telecommunication network, computer

However skeptical or critical one might be, no one wants to be accused of having "eyes that do not see," as Le Corbusier did in his 1923 Towards A New Architecture.

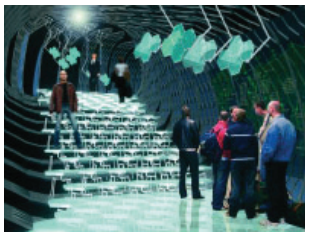
Designer

The design process contains several aspects and several participants. Traditionally it is the genius human being who can create concepts and has the capacity to develop it throughout a pencil and a piece of paper. Traditionally, as an architect, the human being has the main role in the design process. He is the one that takes decisions and determines evolutions throughout the creative process of the production of a design. **Ratio** and **emotion** lead to decisions, either founded or not, which sends the design into a certain direction. Each decision is devoted on the interaction between people and exactly that makes him entirely responsible for the out coming results. Since the era of computers and the rise and growth of information technologies things has changed. Humans and its brain is **completed** by a computer. This computer is frequently considered as emotionless rack strength. The discussion either a computer is capable of feeling emotions is not of relevance within this discussion, but this extra player in the design process plays a bigger role than we first thought. As shown in this thesis, the computer can be applied in several ways.



Ontwerpers

Elektronica



Hybride ruimte

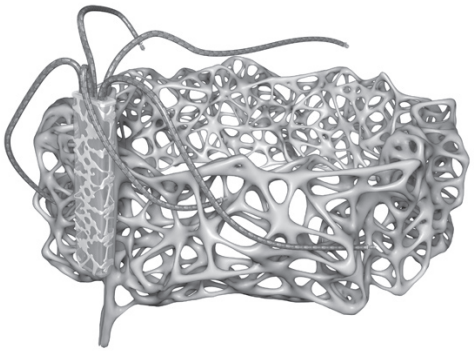
Integration

The most applied method of how the computer is used in an architectural process is the method of the '**proper worker**'. The presentation quality of the computer is explicitly used in which the precise functioning and pure lining is characteristic. The architect designs and the computer draws properly. This is the traditional way in which we architects uses the computers today. Another method is that the computer is used as **a tool** in which he supports the master architect. The computer assists the architects' imagination throughout its calculation strength. When the computer is used as a tool, the architect remains posses of the final responsibility. He decides and also determines if the computer methods must be adapted. He

"Traditionally it is the genius human being who can create concepts and has the capacity to develop it throughout a pencil and a piece of paper."

determines if the result is ok or has to be adjusted. The only power that the computer inhabits, is the role of an executing agency. The architect speaks and the computer calculates. An example of this method is the project that has been discussed under throughout a parametric approach in chapter six. The computer can also be used as **an inspiring tool**. In this method specific data's, throughout a diversity of graphical methods, is used to create datascapes which can work inspiring for developing architecture. When an interesting consequence has been found, these are developed according to traditional methods of architecture. In chapter six we see an example of a project approaches throughout this method. In a final method the computer can act as **a metaphor**. Here the computer forms the base and the air of the architect is minimised. An inversion emerges. Not the computer is the follower, but now the architect becomes the follower, who listens to his boss, the computer. There is a fundamental difference in this method compared to the tool and inspiring method. In the last two methods the architect is the base and is concerned with taking the decisions dealing qualitative and experience qualities. Within the 'divine' method this role is differently. Humans program computers, and determines in this way several things according to personal conviction. However when a language is once confessed, the route is open. The computer calculates and forms. She makes the decisions and delivers the product. Humans make, the architect asks, the computer provides. The last two projects of the chapter practice deals with this method.

Computer as a tool



Computer as a metaphor

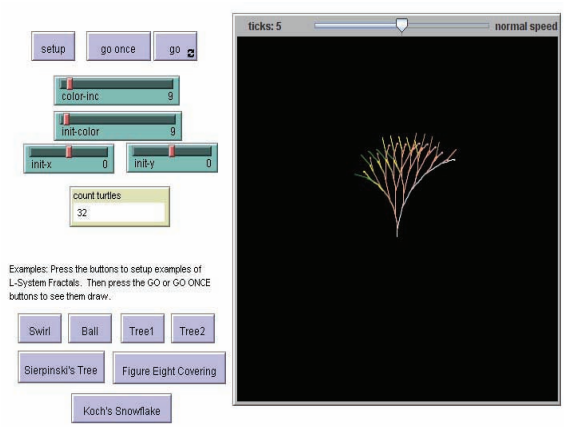
"Humans program computers, and determines in this way several things according to personal conviction."

An important issue must be raised, namely to what extent this power reaches. It is a misconceived to think that a computer is able to work independently. A computer can do nothing else than calculations. Its his only **task**. The idea that a computer can design architecture is misconceived. The computer can do only that do what people, as an interaction medium, tell them to do. The power of the computer during the design process lies therefore not particularly in the creative capacity to take decisions, but lies mainly in the inexhaustible energy and calculation strength through which she can execute certain things. Each task carried out by a computer has once been introduced, considered and created by human influence. The profit is gained especially in the speed by which a computer is able execute things. This feature is especially come to discussion in the optimization project of Michael Hansmeyer analysed in the chapter Practice under project two: approach with a view to feedback. It is for this inexhaustible power that people in the early 70s wee worried and whether this would not lead to the development of intelligence. That computers would continuously optimise itself and become more and more intelligent this way. Many people assumed that the computer would take over power and could completely replace human activity. Although there already specific functions transferred to the computer, it is still a human being that controls the assignment. When the computer is used throughout the design process, not all power comes to the computers' posses. The techniques and methods which she uses, ables people (as an interaction medium) to obtain certain results, for example throughout scripts.

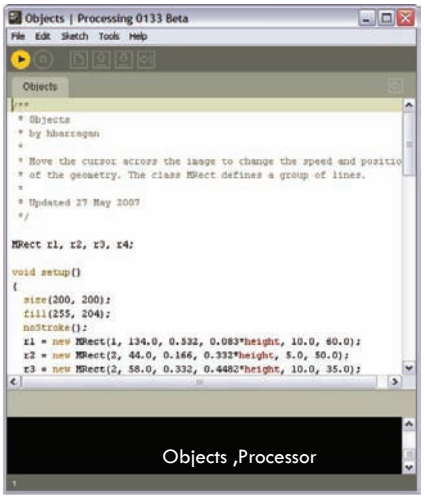
The techniques and applications which were quoted in the previous chapters are all tested by mezelf, on small scale, throughout trial & errors. Like during an **interview** with ir. G. Joosen of ONL I have seen interfaces, scripts written and developed by the office itself, which allows the planner or architect and the property developer or user to adapt and change variable parameters which are immediately executed in the three-dimensional model, designed within the virtual world of the design process. When we **model** in CAD, like for example in 3d Studio Max, we all agree that we create geometries by manipulating topological forms such as a box, a sphere or a torus that we all recognize in the contemporary iconic architecture of the city. During the **development stage** of a building or a master plan we can no longer work without CAD/CAM (ex. Revit) that allows us to continuously carry out changes without having to run through the design process again. Meanwhile we have reached the phase where the discovery of evolutionary principles, for example how certain microworlds

"A computer can do nothing else than calculations. Its his only task. The idea that a computer can design architecture is misconceived.

operate and how principles like L-systems work, not only lead to manage the external characterisations of a building but even moderate the genetic material or DNA of a building. The quoted examples show their usefulness and applicability. **Applications** such as NetLogo and Processor provide the designers the possibility to describe and execute scripts wich implements these systems. Several CAD programmes offer numerous plug-ins on the Internet which are usefull for treating several issues within a design process. In each CAD/CAM your able to create your own plug-in throughout **script editors**.



L-systems Fractals ,NetLogo



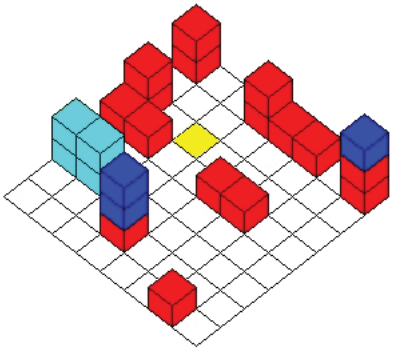
Objects ,Processor

My **graduation project** or graduation project (`science park' at Waterschei, Genk, Belgium) is an excellent possibility to apply a script. However, throughout the urban stage which I have recently passed through, with my knowledge I have conducted I wasn't able to successfully apply these techniques. Of course, during the analytical stage I have been drawing en developing maps continuously. However, the project site is a Brown field of old mines, and so it was nearly impossible to develop useful datescapes. The next stage my final project consists the design of one building. This building will function as an incubator for the science park where spin-offs can rent units, positioned in a certain framework, and extend if they want to or need to. An idea emerges of dividing the three-dimensional framework into **lots** defined by relevant principles throughout a script. Through my thesis research I have encountered

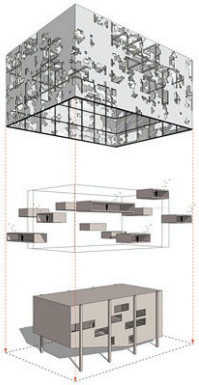
"The profit is gained especially in the speed„by which a computer is able execute things.

some works which are based on arranging volumes according to certain defined principles it seemed a perfect time to apply, the knowledge that I have been gathering during my thesis research, this in practice. However, until now, I haven't reached the desired result to reflect a correct result of this process.

Programma, functions



Algorithm, rules



The 'skin'

At every approach, of the previous projects and my final project, that has been discussed, , the postulated goal has always been determined very clearly. The programme had to be written down exactly so that it could be translated into a relatively easy script. As discussed earlier, the role of the architect contains the fact that he can create a concept and develop it in either a traditional or non-traditional manner. It is therefore the architect who speaks and the computer that calculates. However, between these two actors we miss a link, and that is the person who masters the linguistic of computers. As Daniel Davis indicated and like I have experienced during several solicitations, where the main requirement is computer knowledge, is that the architects have to take notice of these linguistics before the computer programmers take over the design process. But is this true? Do we have to control the linguistics of the computer? Can computer programmers design buildings without the help of architects or otherwise?

Protocol

If we look back to the project I've heard about… we notice that a territorial protocol has been drawn up by R&Sie(n), Benoît Durandin, Alexandra Midal, Laurent Genefort, Gilles

"A protocol is a behavioral agreement, generally in the form of a number of steps."

Schaeffer, Berdaguer & Péjus and François Roustang, with preliminary advice from Bruce Sterling. A `Protocol `is **a behavioural agreement**, generally in the form of a number of steps. And only if all parties behave according to the same agreements it is possible to coordinate their operations. Perhaps the task of the architect lies in this notion. The architect describes the task and the goal, the concept and the development, the programme and the rules. This way he describes **a story**, a body, a building or a biostructure which can therefore be understood by other parties. As so the computer programmer and the computer only execute. The building developer and his user can interact, the architect keeps control concerning the process and intervenes if necessary. If we assume that the technology of construction systems and materials continuously increases, we can assume that such processes not only take place during the design process in the virtual space but would even take place in the real space of the built environment, will the decennial responsibility of the architect than be extended. As such, does the architect becomes responsible for its child? Does the architect has to educate and form his child to be able to live independently in the built environment? With this thought in mind we must reflect, more than ever, on what the future will bring.

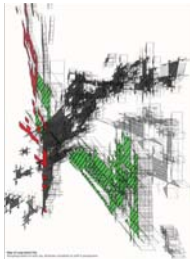
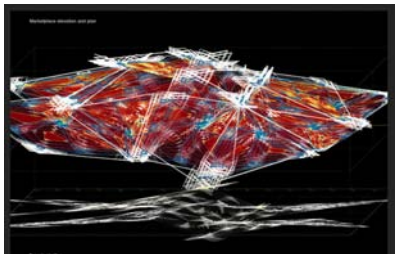
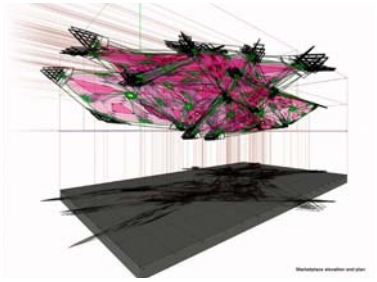
Personal

What people should well understand is that the design of the architect translated into algorithms and described in scripts, just as in the conventional way of designing, is specific for a location. A concept translated into a protocol is therefore not generally applicable to any architectural project. The protocol is a reflection a the designers vision, an urbanist or an architect. Throughout this thesis is proved that all algorithms, every script of each project has been designed for a specific purpose and is certainly not designed to provide a general ideal.

Data

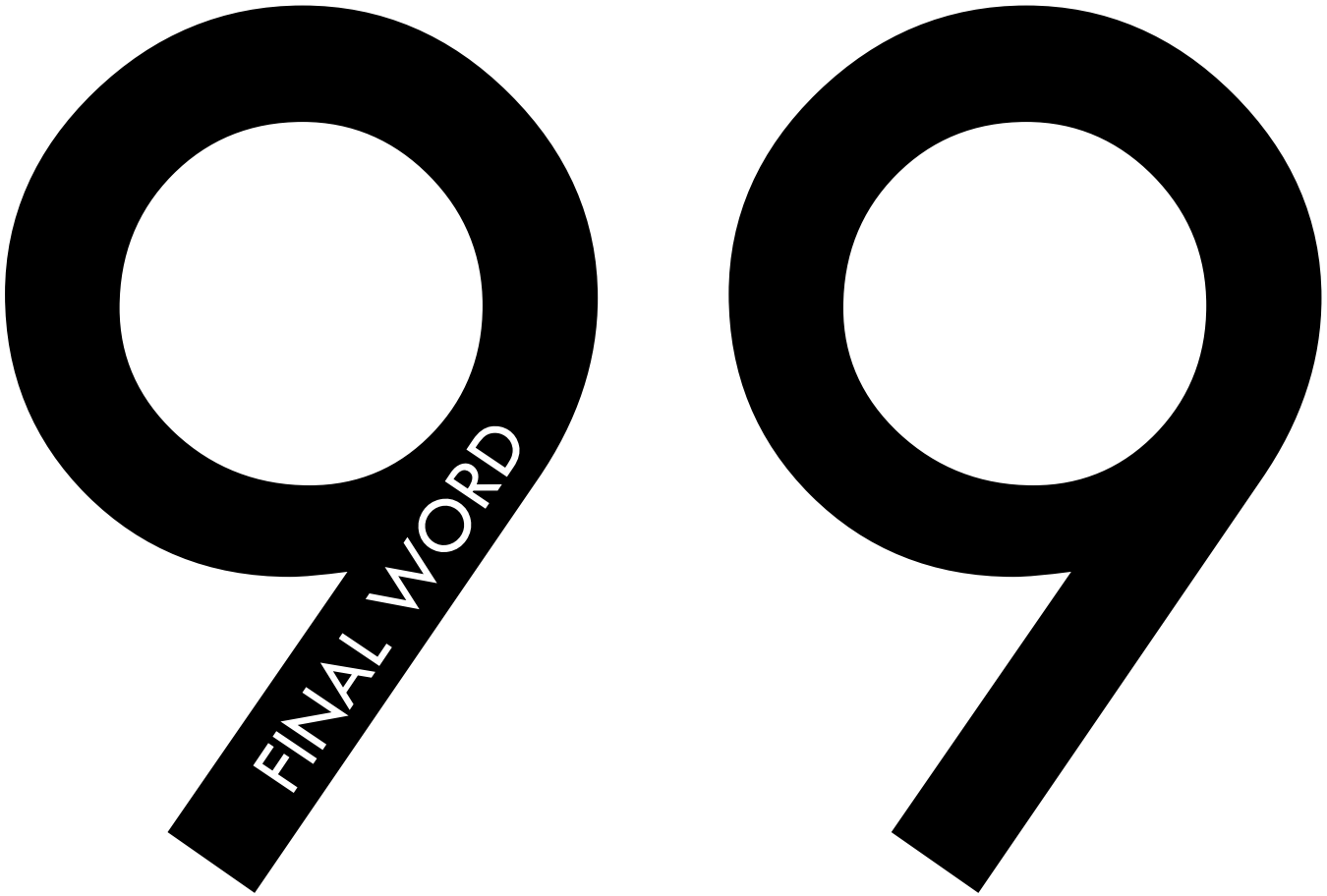
The systems described in this thesis are historical relics, early attempts in linguistics and artificial intelligence. As Stylianos Dritsas refers: their underling structural paradigm is old-school. He suggests we should try to look towards evolutionary systems based on **unstructured models** and query operators (metadata) as in the 'Google' search approach. If we focus further on this aspect, the term **metadata** comes forward. Actually, during this thesis we have continuously been dealing with metadata. For example, a simple 3D object like a box can contain metadata. Because several properties can be given to a box (e.g. name, class,…). In the chapter theory dealing about scripts, I have referred to guidings of script-

"The architect describes the task and the goal, the concept and the development, the programme and the rules.

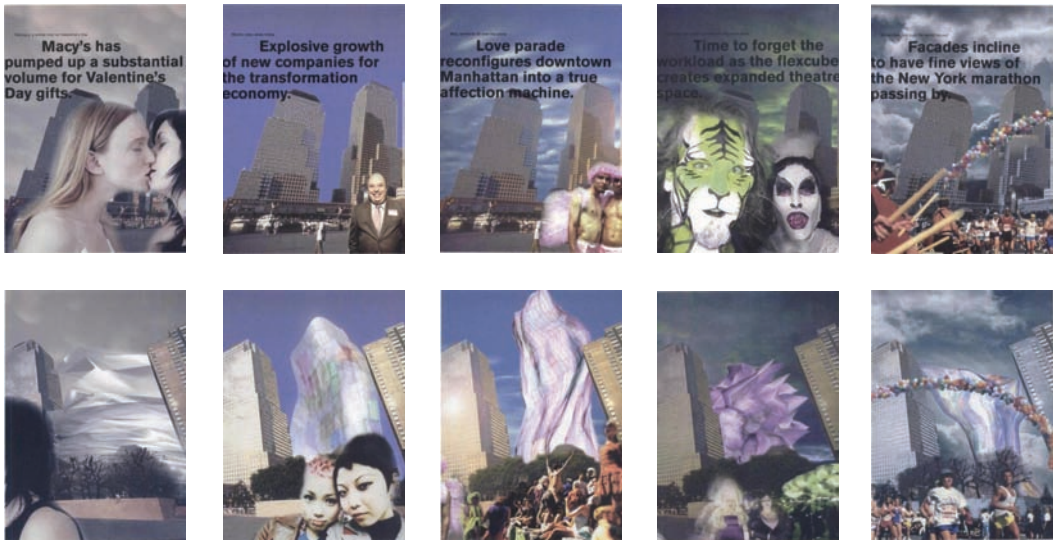
Transformation, Michaël Hansmeyer

editors to get familiar with all these terms. As such, metadata are datas hidden next to each other or behind each other. Therefor, metadata is simply data over data or information of those datas. If we reflect these datas graphically to extract information from for further use in the design process, it is very important to avoid direct formalistic models. In this context MVRDV introduced the term **datescape**, which refers to a visual representation of measurable strengths that has some influence on the concept and development of a design. In the chapter practice we have discussed a project where the site has been approached in this manner. As seen it is possible to superpose several datescapes. As so, a complex spatial envelope is created with frequently contradictory, paradoxical conditions. A space which seems impossible to comprehend throughout our imagination, a digital feeding floor for a new architectural project. The challenge is to avoid that these diagrams of contextual flows and strengths are literally translated in architectural forms. As shown in the project `transformation' by Michael Hansmeyer in his portfolio `Algorithms in Architecture'. In general we should avoid direct formalistic models and focus on pragmatic metric indices of the urban fabric. This conclusion will be the basic thought among others for further research within the urban seminar and architecture.


99
FINAL WORD

A VISION

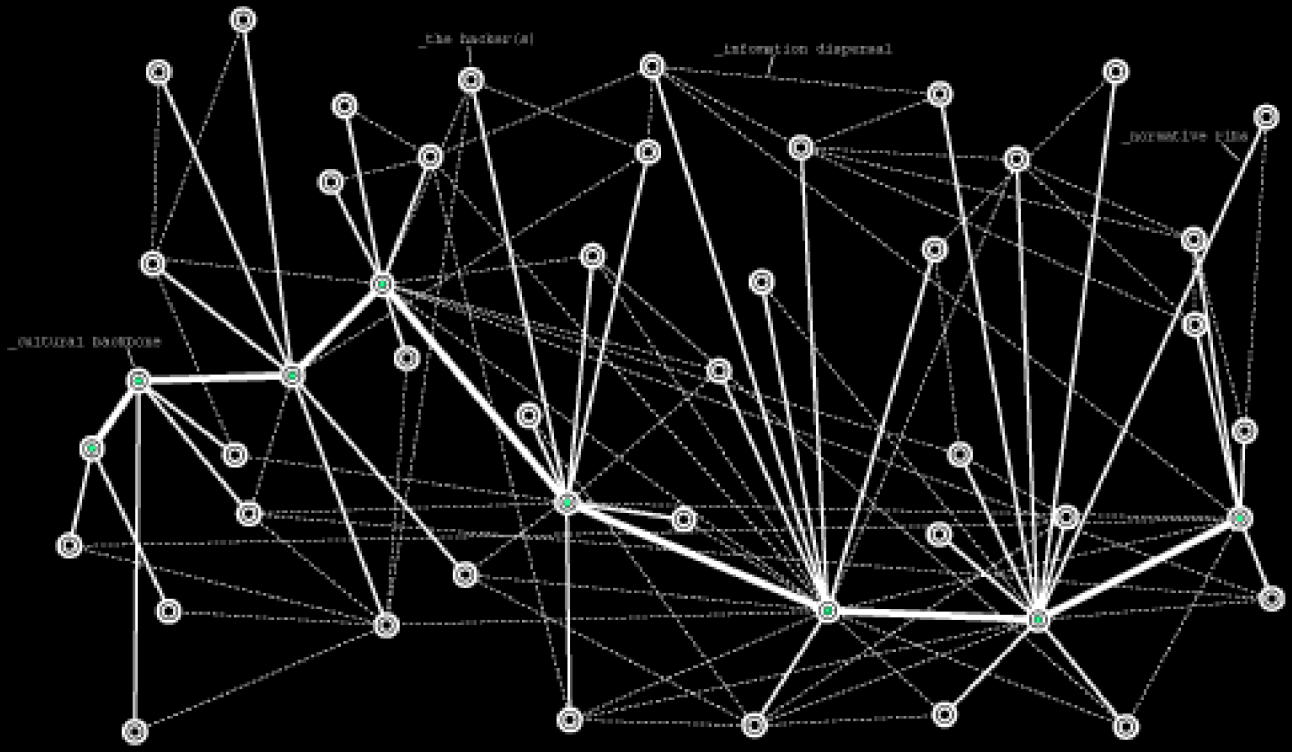
The design of the urban environment becomes a game played by all involved parties. During the design process a game is designed by the architect and played by all relevant parties. During the design of the building, within the 'hybrid space', the game is played by their users, by visitors and by the built environment itself. By playing the game the participants set the parameters. Each actor triggers an array of sensors writing the new data to a database, from where the building picks up the new data and starts reconfiguring itself, in shape, in content, or both in shape and content. Then the new configuration will match to the desired conditions. The building will be in a state of continuous process. The building, consisting of numerous cooperating algorithms, will perhaps behave as an autonomous system. The building will show relational behaviour, always keeping an eye to the neighbouring buildings, always ready to act and react according to the rules of the script.



'Architecture Goes Wild', Kas Oosterhuis

"Urbanism becomes a game being played by its users. By playing the game the participants set the parameters. Then the new configuration will match to the desired conditions."

 The content of this thesis is partly accomplished by the interest of several organizations and persons. The urban seminar at the provincial college in Limburg directed by P. Bongaerts, who gave us the opportunity to experiment discussing this field of research. I also want to thank my promoter Oswald Devisch who has always shown his interests throughout the workshops and guided me through my thesis research. Architectural firm ONL gave me the opportunity to interview ir. G. Joosen discussing several issues of my thesis content. I want to thank E. Vanderfeesten for his cooperation discussing the project 'Confection for the masses in a parametric design of a modular favela structure'; M. Hansmeyer who explained me the five approaches of algorithms in architecture; D. Davis with who I mailed several times discussing his script presented in chapter V; R. Snooks who advised me during exploring several techniques; S. Dritsas who referred to his master thesis and directed me towards new research fields and K. Murphy in name of UN Studio. All of them have cooperated to finish the thesis successfully. My research to information started by surfing on the web. Therefor, I also want to thank B. Gates which has made it possible to gather all data and information on the World wide Web. University libraries Diepenbeek, Hasselt, Leuven en Delft leaded me towards specific books of K. Oosterhuis with 'Architecture goes Wild' and 'Algorithmic Architecture' of Kostas Terzidis.


_the hacker(s)
_information dispersal
_normative ethics
_critical practices

INDEX & SOURCES

03

CONCEPTS

Glossary

LINKS

Papers

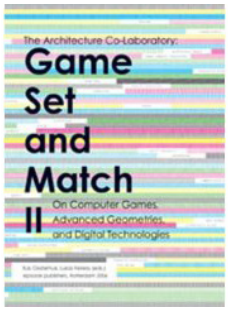
Antoine Picon, z.d., Le temps du cyborg dans la ville territoire, Les Annales de la recherche urbaine nr. 77, z.d.
Guy-Ernest Debord, 'Theory of the Dérive', z.d., z.p.
Marcos Novak, 'Transarchitecture', z.d., z.p.
Marcos Novak, 'Transmitting Architecture: The Transphysical City', z.d., z.p.
Elizabeth Sikiaridi en Frans Vogelaar, 2006, 'Soft Urbanism', Open nr. 11
Troy Innocent, 2003, 'Exploring the nature of electronic space through semiotic morphism', Faculty of Art & Design, Monash University
Axel Kilian, 2000, 'Design Exploration through Bidirectional Modeling of Constraints', Massachusetts Institute of Technology
Stylianos Dritsas, 2004, 'Design Operators', Massachusetts Institute of Technology
Christopher Alexander, 'A city is not a tree', z.d., z.p.
Karl Chu, 'Metaphysics of Genetic Architecture and Computation', z.d., z.p.

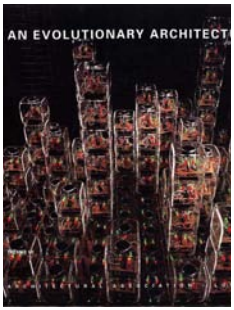
Magazines

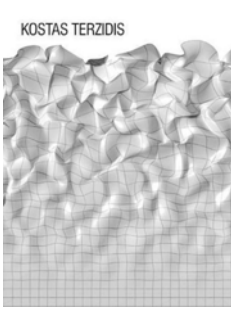
Michael Hensel & Achim Menges & Michael Weinstock, Maart 2006, 'Techniques and Tech nologies in Morphogenetic Design', Architectural Design, z.p.
Christopher Hight & Chris Perry, Oktober 2006, 'Collective Intelligence in Design', Architectural Design, z.p.

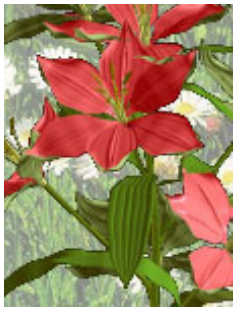
Books

Howard Rheingold, z.d., 'Smart Mobs', The next social revolution, z.p.
John Frazer, 1995, 'An evolutionary architecture', Architectural Assocition
Przemyslaw Prusinkiewicz & Aristid Lindenmayer, 1990, 'The Algorithmic Beauty of Plants', Springer
Kostas Terzidis, 2006, 'Algorithmic Architecture', Architectural Press
Kas Oosterhuis & Lukas Feireiss, 2006, 'GameSetandMatch II', episode publishers
Kas Oosterhuis, 2002, 'Architecture Goes Wild', 010 Publishers
Kas Oosterhuis, 2003, 'Hyperbodies: Toward an E-Motive Architecture', Birkhäuser









Web

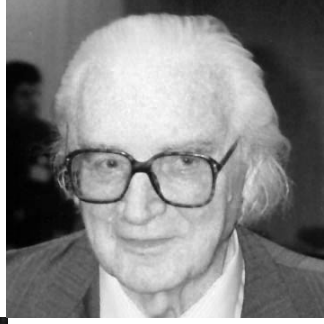
http://users.skynet.be/ludology/
http://www.unrealstockholm.org/wiki/index.php?title=Main_Page
http://www.aaschool.ac.uk/aadrl/
http://www.mat.ucsb.edu/~g.legrady/rsc/algo_arch.html
http://armyofclerks.net/ArchiKluge/index.htm
http://didactiekinf.uhasselt.be/aac/index.php?n=Main.Home
http://www.architektur.abk-stuttgart.de/richter/richter.html
http://gamestudies.org/0501/
http://b.durandin.free.fr/iveheardabout/iha.htm
http://www.kokkugia.com/
http://users.skynet.be/ludology/
http://www.mh-portfolio.com/
http://www.nzarchitecture.com/
http://www.proce55ing.net/
http://www.scriptedbypurpose.net/
http://www.spacesyntax.com/main-nav/home.aspx
http://dritsas.net/
http://www.unstudio.com/
http://www.oosterhuis.nl/
http://www.archi-students.org/forum1.php?id=15
http://www.treddi.com/forum/index.php?showtopic=20677
http://genetic.ming3d.com/GE_FEIDAD3.pdf
http://downlode.org/Etext/Patterns/index.html
http://ncertainties.wordpress.com/

Email

Oswald Devisch: Oswald.Devisch@phl.be
D. Davis: daniel@nzarchitecture.com
G. Joosen: joosen@oosterhuis.nl
Eric Vanderfeesten: info@evanderfeesten.nl
Karen Murphy: k.murphy@unstudio.com
Marc Fornes: marcfornes@hotmail.com
Michael Hansmeyer: michael@mh-portfolio.com
roland snooks: roland@kokkugia.com
Stylianos Dritsas: dritsas@gmail.com

ONL

LE CORBUSIER



MVRDV

NOX

TERZIDIS

REISER &
UMEMOTO

HADID

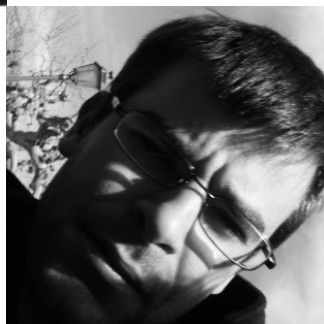
NZ

KOKKUGIA

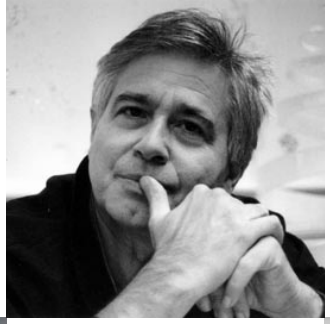
SULLIVAN

MIES VAN
DER ROHE

VANDERFEESTEN

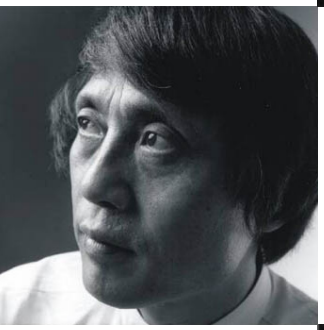


PC



SCHUMI

UN



ANDO

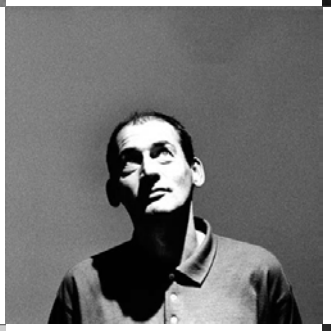
GREG
LYNN

NOVAK

CONSTANT

CHU

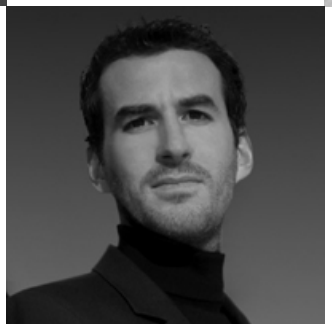
GHERY



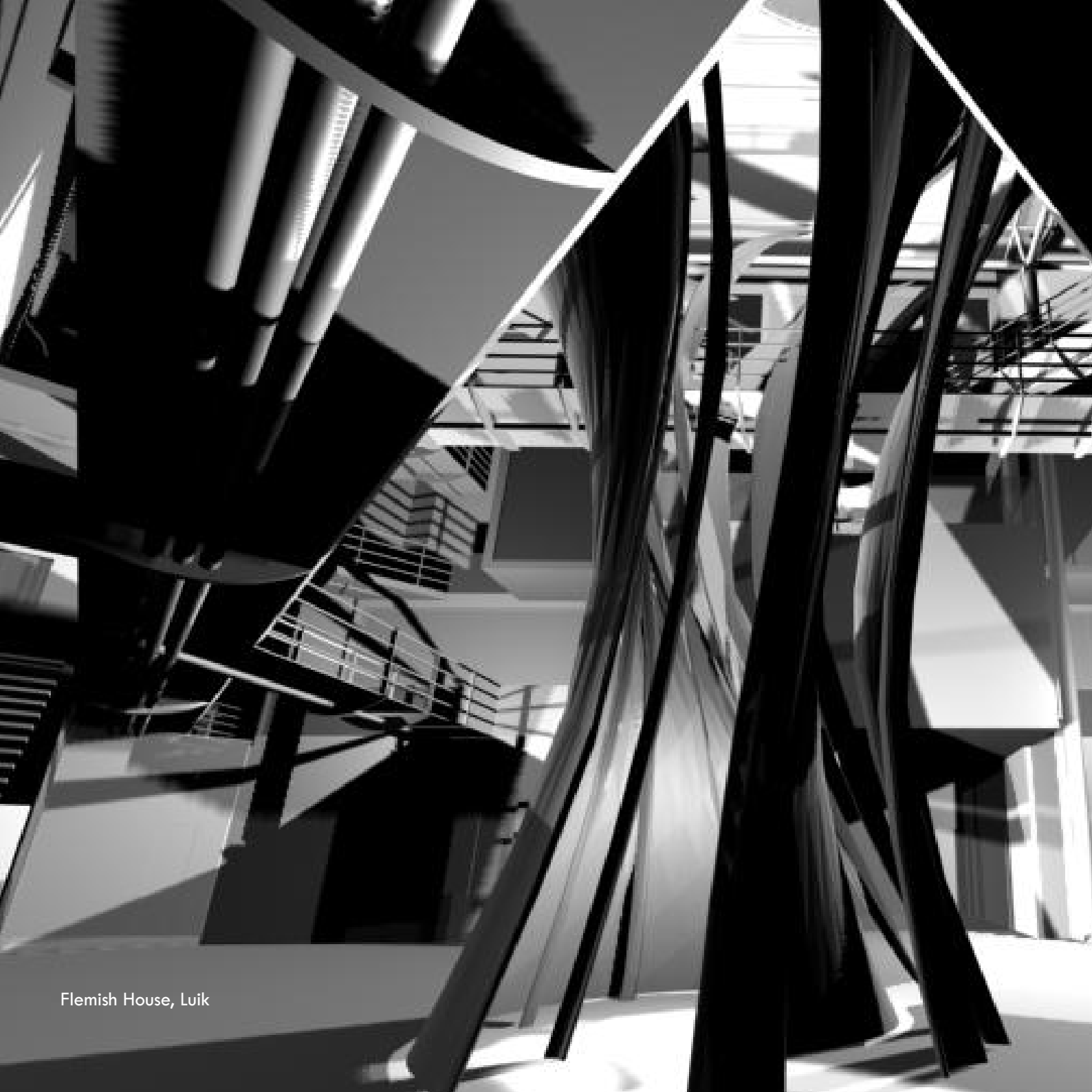
OMA

R&Sie

ACHIM
MENGES



SHUMACHER



Flemish House, Luik

Sueters Bart
Student
PHL
2Ma
Architectuur
2007-2008

Personal information:

- Status: not married
- Nationality: Dutch
- Age: 23 year
- Birthplace: Bonheiden, Belgium
- Parents: John-Paul Sueters & Marjanne Jansse van Noordwijk
- Contact: Veldonkstraat 272 - 3120 Tremelo
- Mail: suetersbart@hotmail.com

http://www.architectennet.com/

The virtual world and the real world live more and more side by side. But how far is this similarity in the designs of architects and urbanists? The reason for this development is preceded by recent history. As a result of a number of technological revolutions, an electronic dimension is created which is described in architectural terms as 'hybrid space'. This space is crossed by electronic infrastructures and networks, where alternative cultural and social fields emerges, which often are not visible from a classical point of view. Thus, claims Antoine Picon, the city for example is developing more and more characteristics of the virtual world of a game. There is no coherence. In this way claims Picon, citizens start to behave more and more as `gamers'. On conferences like 'Game, Set & Match I en II,' organised by K. Oosterhuis, this evolution is examined from the designers point of view. During the design process the game, or architecture , is designed by the architect. This vision forms the underlying thoughts during my thesis research. The title of this script 'design game' refers to the computerscripts that urbanists, architects, engineers and computer programmers develop to create a 'synthetic architecture' which is highly designed within this 'hybrid space'. Computers have been developed and inspired by the fascination of evolutionary principles which takes place in nature. These principles are described according to the logic of algorithms. An algorithm is a process of addressing a problem in a finite number of steps. An architectural design process can often be defined in the same way. With this history, this vision and current technologies some progressive planning- and architectural firms are operative in design practice. ONL (Rotterdam, the Netherlands) apply parametric designing on a manner described as file-to-factory. Space Syntax (London, United Kingdom) use computer-based modelling technique to treat cities and buildings. Throughout all chapters it has become clear that design is a multifaceted process. As such, design tools have often been developed to address isolated issues in design. Through five projects several approaches to different issues are brought to light. To highlight this, a visualization of approaches has been made on which each project can be positioned. Through the study of projects we see that the game, of architecture, is played in the virtual world of the design process. But If the technology of construction systems and materials continuously increases, we can assume that the game, in projects of Kokkugia (Urban Agency) or R&Sie (I've heard about), not only takes place in the virtual world of the design process but even continues into the real world. The question is: will there be a time in which this game is really played or is this vision restricted to what is called an Utopia?