# [De] Coding Architecture

## Open source methods of spatial simulation

## Summary Report

P. Langley

Sept 2013

# Contents

1. Research Question
2. Context and Contribution
3. Methodology
4. Thesis Structure and Case Studies

Appendix
i. diagram of thesis structure
ii. timetable

## 1. Research Question and Context

### Coding Architecture: Open source methods for spatial simulations

Architects are typically consumers of software as products, rather than users of software as technology, and the relationship between computers and architecture can often be fractious – viewed with either complete suspicion or total devotion. In either case the role of the computer is often seen in the same way – to supplement a pre-existing design process rather than to fundamentally alter it. This research project investigates how critical, open source approaches to computational techniques can inform a practice of architecture that is defined by digital methods, rather than simply composed of digital tools.

Commercial design and analysis software mediates both the design process and policy in the built environment as never before. The ability to generate complex digital models of architectural design proposals is rapidly increasing, but critical reflection on the consequences of such advances are not keeping pace. Geometry and data are created in huge quantities during the design of a project and the digital models that this data produces are commonly used to determine a building's supposed future performance, across a range of criteria. But by what means are such software and simulations created and in what ways do their formations influence the design process and contribute to the built outcomes?

My research will examine the ways in which 'code' becomes first 'software' and then 'simulation', and through open source approaches look for the new understandings of 'interface' that are required between the digital and the designers. I will look to address the following questions;

**- How can the use of spatial simulations in architectural design be 'open sourced'?**

**- What new open source methodologies are required and how could they be implemented?**

**- By what process does code 'stabilise' to form digital design and simulation tools?**

**- How is the creation and use of digital tools changed by an open source approach and what new kinds of knowledge production does it allow?**

## 2. Context and Contribution

Many forms of contemporary architectural practice are increasingly characterised by the use of complex, sophisticated software for the design, analysis and production of projects. Previously, software such as CAD has been skeumorphic – mimicking non-computational techniques such as hand drawing – and predominantly providing improvements in the speed of production and organisation of design material. However, current developments in the field of architectural software, such as BIM, represent a significant change in the relationship between architects and the digital software they employ. Single design models can contain full geometric information – at a 'virtual' 1:1 scale – as well as massive amounts of data related to construction cost, environmental performance, energy use and structural stability. Access to this data is typically via a limited selection of proprietary, commercial software packages, the capabilities of which has lead to a focus on tools rather than methods. Furthermore, the extent to which such technology mediates design processes is poorly understood.

The 'open source movement', established through developments in computational technology, provides the background to my research. The software packages commonly used in architectural design are closed and commercial, and although in some case open source alternatives exist, industry wide pressures ensure the status quo is maintained. The commercial benefits available to designers through exploiting the latest features are as significant as the challenges involved in 'breaking from the pack' in order to adopt less popular, and often less feature rich, open source alternatives. Equally huge is the task to overtake major commercial companies in providing innovative software for architects to create 3D models. Furthermore, whilst open source simulation software is available (for example the Energy Plus environmental analysis package), there is not yet a full range that is easily adopted. In this context, the deployment of open source thinking has been at the level of the design outcome, and the free release of 3D models, rather than focused on the development of open source methods of digital design and simulation.

For the purposes of my research, I define digital simulation as the use of software to undertake performance analysis of digital designs where outcomes are evaluated against a predetermined criteria. In this formation, the simulation represents a 'stabilisation' of the software, containing not just the software's own functionality but also the criteria for its success or failure, which are determined externally. The software is a tool and is itself a 'stabilisation' of a once fragile code. Examining this process of 'stabilisation' at each of the different scales it operates at is crucial to understanding the implications of retaining the current 'closed' approach to software and the possibilities of developing more 'open' approaches.

Within this contextual description, the principle contributions of this research project in the field of open source digital simulation for spatial design will be;

**1. To offer an alternative strand of open source architecture and design**

**2.  To provide new, open source methods of spatial design and simulation that can utilise current technological developments rather than replace them.**

**3. The creation of innovative prototype tools for use within this these open source methods.**

**4. To demonstrate the use of these prototype tools for open digital simulations through 'live' design situations.**

## 3. Methodology

*Software as a Field Of Study*

I will take my methodological approach from the emerging field of 'software studies' (Fuller, 2003, 2008). This discipline is concerned with ways in which software is formed, and therefore defined, and is focused on the social production of software and its 'productive capacity' rather than the 'tools' that it produces (Kitchin & Dodge, 2011).  I take the onto-epistemological position that the scientific knowledge against which digital simulations are tested is “...discursively constructed, socially produced *and* materially real...” (Hayles, 2005, p. 209) and my thesis ideals with the ways in which software and code combine to produce such knowledge about spatial design. This will be explored through a series of case studies examining both the ways in which existing software has been formed as well as propositions for new ways of making and prototyping software.

*Coded Traces - genealogical narratives of code*

Software is a 'stabilisation' of code (Kitchin & Dodge, 2011; Mackenzie, 2006) and therefore the code from which that software emerges should represent the 'site' of investigation (Gabrys, 2010). Rather than examining the flow of material through a network of actors (Latour & Woolgar, 1986),  I propose to investigate software by 'following the code' and creating genealogical mappings of its development process that trace its 'unfolding' (Kitchin & Dodge, 2011).

In its idealised form, this would be seen as a wholly quantitative method, taking the database of all code modifications as the data set, analysing and then visualising the changes. While there would be potential to access a development dataset of an instance of open source software, it is highly unlikely (if not impossible) to collect it for closed, commercial software. Therefore a history of the code must somehow be constructed. This step immediately adds a qualitative dimension to the method, and implies that the genealogical mappings can only produce a genealogical 'narrative', that is one of many potential genealogies, rather than a single, definitive description.

In order to write the 'code history', I propose is to take other previously published material related to the software and 'translating' it into code to build a 'teleological historiography' (Kitchin & Dodge, 2011) of the software algorithms that can then be used to map its *genealogical code narratives.*

*Coded simulations - research by hacking*

*"To make a successful hack the hacker needs to keep the power on, keep the flows through the system intact, to keep it functioning as a tool, but reclaiming*

*it, submit it under his will by taming and modulating the flows through the system" (Busch & Palms, 2006, p. 60)*

As I have stated previously, my research is concerned with defining a method to open source the process of digital simulation in spatial design and will develop alternative strategies and tactics in order to do so. I propose a 'research by hacking' approach that takes 'hacking' (rather than 'cracking') as a creative, rather than destructive method (Busch & Palms, 2006; Raymond, 2001)

'Hacking' as method goes beyond 'customisation' and instead seeks to fundamentally alter the designed characteristics of something but without re-making it completely. It is a 'modifying culture' (Busch & Palms, 2006) that is explicit in its recognition of the wider 'ecology' within which it exists. Instead of attempting to design completely new systems of simulation, I will use a 'research by hacking' approach to identify and modify the system by which spatial simulations can be 'hacked'.

I propose the design and use of specific, open source prototype software tools for simulation as well as investigating the nature of code as a 'design material'. This will include the iterative prototyping of code and software, and will utilise student projects of the MArch and MAAD 'Open Data' Design Studio, of which I am co-lead. The studio will act as an environment for testing and participatory co-design during hacking sessions.

## tools and techniques

I will adopt a range of coding languages and techniques as part of this research, but two key tools will be *Github* (www.github.com) *and Processing* (www.processing.org), both of which are open source and cross-platform.

*Github* is a versioning platform for 'social coding', based on the popular *Git* versioning software. As with much versioning software, *Github* creates structured databases of changes to code within shared projects, in the form of 'branches', 'merges' and 'commits'. While *Github* itself provides some analysis tools, the structured data sets that it produces can be extracted for custom analysis and dynamic visualisation using software such as *Gephi*.

*Processing* is a well known and popular coding platform that has its origins in data visualization. Its use has now grown beyond this field, in part due to its ease of use and thorough documentation. Another aspect of this software is the wide availability of libraries that can be used to extend its core functionality. I will use *processing* as the principle coding tool for prototyping.

# 4. Thesis structure and Case Studies

My research is located within the long-standing tradition of the 'open source movement', and its use in the field of architectural design. From this situated position, the thesis is structured around three principle, thematic strands, each one addressing 'code' at a specific 'scale';

**Section 1. Situating Simulations**

**Section 2. Traces of Codes**

**Section 3. Virtual Interfaces**

Each section will include an exploration of the theories associated with its topic and will be supported by a specific case study, described below.

## Situating Simulations

Digital simulations are aggregations of stabilised code that display agency by making material their virtual worlds. This section will investigate the ways in which scientific knowledge is “...discursively constructed, socially produced and materially real...” (Hayles, 2005, p. 209) and the relevance of this statement to spatial simulation and architectural design, and will address the following;

**- Defining constructive and realist ontologies with regard to scientific knowledge (Latour & De Landa)**

**- The use of simulation as a means of producing scientific knowledge (Haraway, Hayles, Barad, Stengers)**

**- The role of narrative in an autopoietic understanding of simulation (Haraway, Hayles & Varela)**

**- Architectural simulation and specifically the techniques of Space Syntax as 'coded space' (Kitchen and Dodge, Hillier et al)**

**Case Study 1 - *GENEALOGICAL SOFTWARE MAPPING OF 'SPACE SYNTAX' SIMULATION TECHNIQUES***

> This case study will create a 'genealogical trace' of the development process that Space Syntax techniques of 'axial mapping' and 'visibility' mapping have undergone since their inception in the late 1970s. This case study will use as its source material the extensive record of peer reviewed material that supports the Space Syntax research project. By adopting a genealogical approach, and 'following the code', the intention is to map through translation and codification the capabilities and functionality of each technique and therefore provide a critical narrative around its construction and contemporary use.

## Traces of Codes

Code is messy and fragile, liable to collapse - always on the 'verge of disappearance'  (Mackenzie, 2006). It exists in its own 'ecosystem' and, if not tended to and nurtured will decay as less and less people remain familiar with its structure and purpose. But code is also social with multiple authorship and the more people that contribute, the less the code is understood (Ullman, 2013). Code exists as an interlayer between humans and machine. It must be 'readable' by both, but is not necessarily understood by either. The description of a code's abstraction and stabilisation into software is also the process by which it gains agency.  By what means is this agency realised and what effect does code have in spatial formations?

**- [In]stabilisations of code (Fuller, Hayles, Kitchin and Dodge, Mackenzie, Ullman)**

**- Spatial stabilisations of Code(Constant, Hackitectura, Kitchin and Dodge, Haque and Fuller)**

**Case Study 2 – *OPEN DATA STREAMS FOR DATA DRIVEN DESIGN***

> *Creating 'open', publicly accessible streams of computational design data geometry and simulation data, specifically environmental data*
>
> This case study is about the design and creation of 'open streams' of design data for public access. It will explore the issues around interoperability through the use of novel file types and forms of data and will use twitter as the means of aggregation and distribution. One of the outputs of this

will be a software plug in for a 3D modelling package. This new technological innovation will be tested and demonstrated through a short series of student workshops.

*Technical aspects of this case study have been developed as part of my @simulationBot pilot project. This was used and tested as part of a workshop in environmental design in July 2013with students form SSoA.*

## Virtual Interfaces

The pervasive nature of contemporary digital technology – coded products, spaces and infrastructures (Kitchin & Dodge, 2011) – is such that the question of 'interface' has moved beyond the traditional model of HCI (Human Computer Interface) (Fuller, 2003). Instead of sitting outside of the system looking across a screen, we constantly re-make our relationships with computational technology and in the process we are creating dynamic, topological, post-human formations (Braidotti, 2013; Haraway, 1991). At the same time, there is an apparent 'de-materlialisation' of technology from *our* everyday lives. This is a process driven by the ubiquity of computing which creates environments so rich in this technology that we no longer make the distinctions between the digital and the non-digital (Shepard, 2011) Our true interface with the materials of digital technology is only revealed through the waste left behind after its manufacture (Gabrys, 2010).

This is the new paradigm for our 'interface' with digital technology. What are the implications of these issues with regard to new types of computational tools for spatial design?

**- Post humanist futures (Braidotti, Haque, Haraway, Hayles)**

**- Digital Materiality (Constant, Gabrys, Hackitectura, Hayles, Kitchin and Dodge, Sheppard)**

**Case Study 3 - *@SIMBOT AS DIGITAL COMPANION***

*A personal, customisable, 'companion app' capable of 'reading' to dynamic data streams*

This case study creates ways in which the design data streams developed in case study 2 can be used as design and teaching tools. It will focus on the design and use of a prototype 'app' that is capable of navigating and 'interpreting' data feeds, as described above. The app will be designed in such a way as to allow customisation and for the user to create their own 'companion app'. This new technological innovation will be tested and demonstrated through a short series of student workshops.

*Technical aspects of this case study have been developed as part of my @simulationBot pilot project. This was and tested as part of a workshop in environmental design in July 2013, with students form SSoA.*

*Outline Chapter Structure*

**Section 0. Open source**

**Chapter 1. Narratives of Divergent Open Source and Architecture**

**Chapter 2. Hacking as Methodology**

**Section 1. Situating Simulations**

**Chapter 3. Constructing Knowledge and spatial formations of simulation**

**Chapter 4. Simulation and Narrative**

**Case Study 1 - *GENEALOGICAL SOFTWARE MAPPING OF 'SPACE SYNTAX' SIMULATION TECHNIQUES***

**Section 2. Traces of Codes**

**Chapter 5. [In]stabilisations of code**

**Chapter 6. Spatial stabilisations of Code**

**Case Study 2 – *OPEN DATA STREAMS FOR DATA DRIVEN DESIGN***

**Section 3. Virtual Interfaces**

**Chapter 7.  Post Humanist futures**

**Chapter 8.  Digital Materiality**

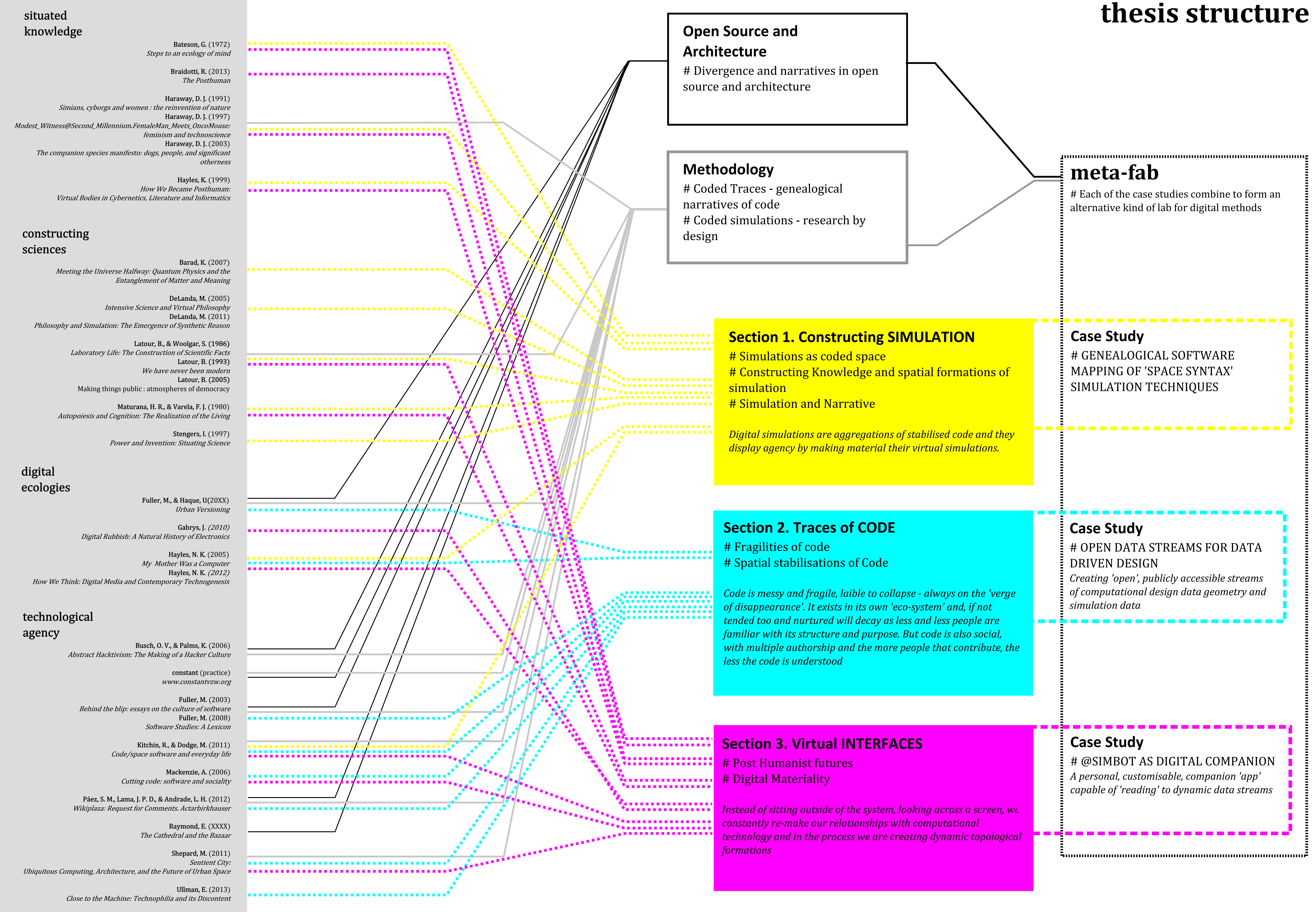
**Case Study 3 - *@SIMBOT AS DIGITAL COMPANION***

## Appendix

i. diagram of thesis structure

ii. timetable

# thesis structure



situated knowledge

Bateson, G. (1972)
*Steps to an ecology of mind*

Braidotti, R. (2013)
*The Posthuman*

Haraway, D. J. (1991)
*Simians, cyborgs and women : the reinvention of nature*
Haraway, D. J. (1997)
*Modest_Witness@Second_Millennium.FemaleMan_Meets_OncoMouse: feminism and technoscience*
Haraway, D. J. (2003)
*The companion species manifesto: dogs, people, and significant otherness*

Hayles, K. (1999)
*How We Became Posthuman: Virtual Bodies in Cybernetics, Literature and Informatics*

constructing sciences

Barad, K. (2007)
*Meeting the Universe Halfway: Quantum Physics and the Entanglement of Matter and Meaning*

DeLanda, M. (2005)
*Intensive Science and Virtual Philosophy*
DeLanda, M. (2011)
*Philosophy and Simulation: The Emergence of Synthetic Reason*

Latour, B., & Woolgar, S. (1986)
*Laboratory Life: The Construction of Scientific Facts*
Latour, B. (1993)
*We have never been modern*
Latour, B. (2005)
Making things public : atmospheres of democracy

Maturana, H. R., & Varela, F. J. (1980)
*Autopoiesis and Cognition: The Realization of the Living*

Stengers, I. (1997)
*Power and Invention: Situating Science*

digital ecologies

Fuller, M., & Haque, U(20XX)
*Urban Versioning*

Gabrys, J. *(2010)*
*Digital Rubbish: A Natural History of Electronics*

Hayles, N. K. (2005)
*My Mother Was a Computer*
Hayles, N. K. *(2012)*
*How We Think: Digital Media and Contemporary Technogenesis*

technological agency

Busch, O. V., & Palms, K. (2006)
*Abstract Hacktivism: The Making of a Hacker Culture*

constant (practice)
*www.constantvzw.org*

Fuller, M. (2003)
*Behind the blip: essays on the culture of software*
Fuller, M. (2008)
*Software Studies: A Lexicon*

Kitchin, R., & Dodge, M. (2011)
*Code/space software and everyday life*

Mackenzie, A. (2006)
*Cutting code: software and sociality*

Páez, S. M., Lama, J. P. D., & Andrade, L. H. (2012)
*Wikiplaza: Request for Comments. Actarbirkhauser*

Raymond, E. (XXXX)
*The Cathedral and the Bazaar*

Shepard, M. (2011)
*Sentient City: Ubiquitous Computing, Architecture, and the Future of Urban Space*

Ullman, E. (2013)
*Close to the Machine: Technophilia and its Discontent*

**Open Source and Architecture**
# Divergence and narratives in open source and architecture

**Methodology**
# Coded Traces - genealogical narratives of code
# Coded simulations - research by design

**meta-fab**
# Each of the case studies combine to form an alternative kind of lab for digital methods

**Section 1. Constructing SIMULATION**
# Simulations as coded space
# Constructing Knowledge and spatial formations of simulation
# Simulation and Narrative

*Digital simulations are aggregations of stabilised code and they display agency by making material their virtual simulations.*

**Case Study**
# GENEALOGICAL SOFTWARE MAPPING OF 'SPACE SYNTAX' SIMULATION TECHNIQUES

**Section 2. Traces of CODE**
# Fragilities of code
# Spatial stabilisations of Code

*Code is messy and fragile, laible to collapse - always on the 'verge of disappearance'. It exists in its own 'eco-system' and, if not tended too and nurtured will decay as less and less people are familiar with its structure and purpose. But code is also social, with multiple authorship and the more people that contribute, the less the code is understood*

**Case Study**
# OPEN DATA STREAMS FOR DATA DRIVEN DESIGN
*Creating 'open', publicly accessible streams of computational design data geometry and simulation data*

**Section 3. Virtual INTERFACES**
# Post Humanist futures
# Digital Materiality

*Instead of sitting outside of the system, looking across a screen, we constantly re-make our relationships with computational technology and in the process we are creating dynamic topological formations*

**Case Study**
# @SIMBOT AS DIGITAL COMPANION
*A personal, customisable, companion 'app' capable of 'reading' to dynamic data streams*

# Timetable

| | Year 2 | | | | | | | | | | | | Year 3 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 2013 | | | 2014 | | | | | | | | | | | | 2015 | | | | | | | | |
| | October | November | December | January | February | March | April | May | June | July | August | September | October | November | December | January | February | March | April | May | June | July | August | September |
| **Case Study 1** | | | | | | | | | | | | | | | | | | | | | | | | |
| Collect source material | ⬛ | ⬛ | ⬛ | | | | | | | | | | | | | | | | | | | | | |
| Translate and encode data | | | | ⬛ | ⬛ | ⬛ | | | | | | | | | | | | | | | | | | |
| Build database | | | | | | | ⬛ | ⬛ | ⬛ | | | | | | | | | | | | | | | |
| Create genealogical mappings | | | | | | | | | | ⬛ | ⬛ | ⬛ | | | | | | | | | | | | |
| Write-up | | | | | | | | | | | | | ⬛ | ⬛ | ⬛ | ⬛ | ⬛ | ⬛ | ⬛ | ⬛ | ⬛ | ⬛ | ⬛ | ⬛ |
| **Case Study 2** | | | | | | | | | | | | | | | | | | | | | | | | |
| 'disassemble' existing @simulationBot codes for reuse | ⬛ | | | | | | | | | | | | | | | | | | | | | | | |
| Develop functionality | | ⬛ | | | | | | | | | | | | | | | | | | | | | | |
| Prototype and testing | | | ⬛ | ⬛ | ⬛ | | | | | | | | | | | | | | | | | | | |
| Hack session 1 | | | | | | 🟥 | | | | | | | | | | | | | | | | | | |
| Reflection and review | | | | | | | ⬛ | ⬛ | | | | | | | | | | | | | | | | |
| Prototype and testing | | | | | | | | ⬛ | ⬛ | | | | | | | | | | | | | | | |
| Hack session 2 | | | | | | | | | | 🟥 | | | | | | | | | | | | | | |
| Reflection and review | | | | | | | | | | | ⬛ | ⬛ | | | | | | | | | | | | |
| Write up | | | | | | | | | | | | | ⬛ | ⬛ | ⬛ | ⬛ | ⬛ | ⬛ | ⬛ | ⬛ | ⬛ | ⬛ | ⬛ | ⬛ |
| Potential additional hacking sessions | | | | | | | | | | | | | | 🟥 | | | | 🟥 | | | | | | |
| **Case Study 3** | | | | | | | | | | | | | | | | | | | | | | | | |
| 'disassemble' existing @simulationBot codes for reuse | ⬛ | | | | | | | | | | | | | | | | | | | | | | | |
| Develop functionality | | ⬛ | | | | | | | | | | | | | | | | | | | | | | |
| Prototype and testing | | | ⬛ | ⬛ | ⬛ | | | | | | | | | | | | | | | | | | | |
| Hack session 1 | | | | | | 🟥 | | | | | | | | | | | | | | | | | | |
| Reflection and review | | | | | | | ⬛ | ⬛ | | | | | | | | | | | | | | | | |
| Prototype and testing | | | | | | | | ⬛ | ⬛ | | | | | | | | | | | | | | | |
| Hack session 2 | | | | | | | | | | 🟥 | | | | | | | | | | | | | | |
| Reflection and review | | | | | | | | | | | ⬛ | ⬛ | | | | | | | | | | | | |
| Write up | | | | | | | | | | | | | ⬛ | ⬛ | ⬛ | ⬛ | ⬛ | ⬛ | ⬛ | ⬛ | ⬛ | ⬛ | ⬛ | ⬛ |
| Potential additional hacking sessions | | | | | | | | | | | | | | 🟥 | | | | 🟥 | | | | | | |