[de]Coding Architecture:
Open source methods for spatial simulations

- Methods not tools
- Open source

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

Situating Simulations

- Constructing Knowledge and spatial formations of simulation
- Simulation and Narrative

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

Traces of Code

- [In]stabilisations of code
- Spatial stabilisations of Code

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?

Virtual Interfaces

- Post Humanist futures
- Digital Materiality

Other ideas of interface

The pervasive nature of contemporary digital technology – coded products, spaces and infrastructures (Kitchin and 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 (Haraway 1991; Braidotti 2013). 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?

@simulationBot

- De-stabilisation of code
- research by hacking

This project attempts a critical engagement with techniques of spatial simulation in architectural design. A 'bot' is proposed that can collate, filter, interpret and re-present live design work. The prototype version of the @simulationBot is a twitter creature it 'feeds' on the vast amount of performance data that is generated.

This version of the @simulationBot is designed to sit within existing modes of design, iteration, simulation and evaluation rather than replacing them and acts as a 'bolt-on' to a pre-conceived simulation workshop.

The work flow of the workshop was as follows;

- Create simple models of design problem using Sketch Up

- Import the model into Processing, using Anar+ library

- Apply parametric controls with a Processing code

- Run energy simulations directly from Processing using Energy Plus

- Evaluate results and modify design accordingly

For the prototype @simulationBot I created the following tools to hack this work flow;

- A custom Sketch Up plugin using rubyscript to export simple models to .obj file format. The exported files included geometry as well materials and topological data suitable for use in environmental simulation software.

- Code templates for Processing that would automatically send tweets of the students design data and images when they ran a simulation.

- simBot, written using Processing, that 'picked up' and re-interpreted the data of each student's design
The code embedded in the processing sketches tweeted an image of the design at the time of the simulation, as well as a series of 'hash tags' that the @simulationBot could use to identify the designer as well as locate the performance data that was produced and shared using a Dropbox folder.